VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Nástroj pro sdílenou pracovní plochu a vzdálenou výuku

Tool for Desktop Sharing and Remote Teaching

2013

Bc. Martin Prokeš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Martin Prokeš**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Nástroj pro sdílenou pracovní plochu a vzdálenou výuku
Tool for Desktop Sharing and Remote Teaching

Zásady pro vypracování:

1. Seznamte se s problematikou sdílené výukové plochy v prostředí internetu.
2. Nastudujte funkce a možnosti existujících řešení, extrahujte základní funkcionality a nabízené možnosti použití.
3. Sestavte seznam požadavků pro systém provozovaný na internetu, který umožní tutorovi výklad v reálném čase s možnostmi sdílené pracovní plchy, přenosu audia a videa. Systém bude obsahovat management studentů, jejich přihlašování na výukové lekce, formulování dotazů a práce ve sdíleném pracovním prostoru. Tutor bude mít možnost připravit harmonogram lekce, který lze na základě aktuální situace ve výuce dynamicky modifikovat.
4. Implementace bude orientována na platformu Microsoft Windows Vista a Windows 7.
5. Proveďte analýzu návrh a implementaci tohoto řešení s použitím technologií .NET.
6. Proveďte testování a stanovte minimální požadavky na technické, programové a síťové vybavení a nastavení, nezbytné pro chod celé aplikace.
7. Zhodnoťte dosažené výsledky v porovnání s konkurenčními projekty a navrhněte další možnosti rozšíření.

Seznam doporučené odborné literatury:

[1] KOMENDA, Martin . Autorský nástroj Ozvučená prezentace [online]. [s.l.], 2007. 51 s. Bakalářská práce. Masarykova univerzita. Dostupné z WWW: <http://is.muni.cz/th/98951/fi_b/>
[2] GRACÍK, Martin. Sdílená pracovní plocha [online]. [s.l.], 2008. 22 s. Bakalářská práce. Masarykova univerzita. Dostupné z WWW:
<http://is.muni.cz/dok/rfmgr.pl?furl=%2Fth%2F143087%2Ffi_b%2F;info=>
[3] Adobe Connect [online]. 07-14-2009 [cit. 2011-05-04]. Dostupné z WWW:
<http://www.adobe.com/products/adobeconnect.html>
[4] Webex [online]. c2011 [cit. 2011-05-04]. Dostupné z WWW: <http://www.webex.com/>
[5] Software as a Service: Strategic Backgrounder : Software & Information Industry Association [online]. Washington, D.C : [s.n.], 2001 [cit. 2011-05-04]. Dostupné z WWW: <http://www.siia.net/estore/pubs/SSB-01.pdf>
[6] Microsoft /web [online]. c2011 [cit. 2011-05-04]. Microsoft Silverlight. Dostupné z WWW:
<http://www.microsoft.com/cze/web/silverlight/>
[7] MSDN Library [online]. 2009 [cit. 2011-05-04]. Windows Mobile. Dostupné z WWW:
<http://msdn.microsoft.com/en-us/library/bb847935.aspx>
[8] Microsoft Office [online]. c2011 [cit. 2011-05-04]. Microsoft PowerPoint. Dostupné z WWW:
<http://office.microsoft.com/en-us/powerpoint/>
[9] VoIP-info [online]. 2011, last updated 2011-05-23 [cit. 2011-05-23]. Dostupné z WWW: <http://www.

voip-info.org/>

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Radoslav Fasuga, Ph.D.**

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že tato práce je mým původním autorským dílem vypracovaným samostatně. Všechny zdroje, prameny a literaturu, které jsem použil a z nichž jsem čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

V Ostravě dne 6. 5 2013 .....

Jméno Příjmení

Děkuji vedoucímu diplomové práce Ing. Radoslavu Fasugovi, Ph.D. za odbornou pomoc
a všechny cenné rady při zpracování této práce.

Abstrakt

Tato diplomová práce se zabývá návrhem a implementací systému pro pořádání webových seminářů. V této práci navazuji na svou bakalářskou práci, kterou dále rozvíjím a optimalizuji. Soustředím se na stěžejní část celého systému, kterou je komunikace a přenos dat mezi jednotlivými klienty ve webovém semináři. Je zde popsaná problematika a úskalí spojené s pořádáním webinářů v prostředí Internetu. Dále je zde popsána implementace vlastního komunikačního protokolu FBP a implementace systému pro pořádní webových seminářů ForceB. Systém je postaven na technologiích .NET od společnosti Microsoft.

Klíčová slova: e-learning, webinář, klient-server, protokol, ForceB, komunikace, RTP, .NET, C#, ASP.NET, cloud computing, Windows Azure

Abstract

This thesis describes the design and implementation of a system for organizing webinars. In this work, I follow my bachelor thesis, which was further developed and optimized. Concentrate on the core part of the system, which is communication and data transfer between clients in the web seminar. It also describes problems and pitfalls associated with organizing webinars on the Internet. Work then deals with implemention of own communication protocol FBP and implementation of a system for proper webinars ForceB itself. The system is built on top of .NET technology from Microsoft.

Keywords: e-learning, webinar, klient-server, protokol, ForceB, communication, RTP, .NET, C#, ASP.NET, cloud computing, Windows Azure

Seznam použitých zkratek a symbolů

API	–	Application programming interface
ASP	–	Active Server Pages
BLOB	–	Binary Large Object
BYE	–	odhlašovací paket
Bandwidth	–	Šířka přenosového pásma
CESNET	–	Czech Education and Scientific NETwork
CNAME	–	canonical name, kanonické jméno, identifikátor
FTP	–	File Transfer Protocol
G.721	–	standart zvukového kodeku
G.722	–	standart zvukového kodeku
G.728	–	standart zvukového kodeku
GUI	–	Graphic User Interface (grafické uživatelské rozhraní)
H.261	–	standart audio-video kodeku
H.263	–	standart audio-video kodeku
H.264	–	standart audio-video kodeku
H.323	–	standart audio-video kodeku
HTTP	–	HyperText Transfer Protocol
ICETA	–	International Conference on Emerging eLearning Technologies and Applications
IEEE	–	The Institute of Electrical and Electronics Engineers
IPTV	–	Internet Protocol TeleVision
ISO/OSI	–	standardizovaný referenční model
JPEG	–	standartní metoda ztrátové komprese
MJPEG	–	standartní metoda ztrátové komprese
MPEG	–	ztrátový komprimační datový formát
MPEG-1	–	ztrátový komprimační datový formát
MPEG-2	–	ztrátový komprimační datový formát
MPEG-4	–	ztrátový komprimační datový formát
Multicast	–	vícesměrové vysílání
PAAS	–	Platform as a Service
pay as you go	–	Neboli kolik toho uživatel spotřebuje, tolik zaplatí
QOS	–	Kvalita služby
Retransmise	–	Opakovaný přenos

RFC3550	–	definice organizace IETF pro použití RTP/RTCP protokolu
RTP	–	Realtime Transfer Protocol, protokol pro přenášení dat v reálném čase
RTCP	–	Realtime Transfer Control Protocol, protokol pro kontrolu přenášení dat v reálném čase
TCP/IP	–	sada protokolů pro komunikaci v počítačové síti
SAAS	–	Software as a Service
SQL	–	Structured Query Language
UDP	–	User Datagram Protocol, nespolehlivý, nespojovaný přenosový protokol
Unicast	–	vysílání „jeden k jednomu“
XAML	–	Extensible Application Markup Language
Webinář	–	Webový seminář

Obsah

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Po celém světě můžeme pozorovat ohromný vzestup poptávky po multimediálních službách. Zejména služby jako IPTV, VoiceIP, videokonference, webináře a podobně, jsou nyní na velkém vzestupu. Tyto služby bylo dříve nemožné provozovat v prostředí Internetu. Dnes s rozvojem Internetu se tyto technologie dostávají stále více do popředí zájmu a s nimi i vývoj nových standardů a protokolů pro přenos multimediálních dat.

Právě přenos multimediálních dat v reálném čase spolu s implementací výukového nástroje pro pořádání webinářů je předmětem mé diplomové práce. Tato práce navazuje na mou bakalářskou práci [1], kterou dále rozvíjí a optimalizuje. Výsledkem bakalářské práce je komplexní systém umožňující pořádat webové semináře v prostředí internetu a univerzitní sítě. Primárním cílem diplomové práce bylo zaměřit se na samotný motor celého systému, kterým je přenos dat. Jedná se o klíčovou část celého systému. Právě tato část byla největší slabinou původního systému, z důvodu omezeného počtu připojených účastníků. Motivací bylo navrhnout vlastní komunikační protokol, jenž by umožnil pořádat semináře až pro stovky účastníků.

Úvodní část této práce se věnuje charakteristice webových seminářů. Popisuje možnosti této technologie a její přínos ve výuce. Druhá část představuje současné systémy pro pořádání webových seminářů a jejich klíčové vlastnosti. Dále je věnovaná část problematice přenosu dat v Internetu a možnostem komunikace mezi velkým množstvím klientů. Následující část této práce je věnována vývoji vlastního komunikačního protokolu ForceB. Je zde popsána jeho architektura, způsob zapouzdření dat, struktura záhlaví paketů, způsob směrování dat ve ForceB síti a problematika související s přenosem dat pomocí UDP protokolu. V dalších kapitolách je rozebrána implementace jednotlivých částí celého systému. Věnuje se implementací FBP protokolu a následně implementací samotného systému pro pořádání webinářů postaveném nad tímto protokolem. V práci jsou také shrnuty zkušenosti s cloud computingem a reálným nasazením tohoto systému na Microsoft Windows Azure. Na závěr jsou popsány a ukázány jednotlivé aplikace celého systému.

Tento projekt se účastnil 10. ročníku mezinárodní konference Emerging eLearning Technologies & Applications (ICETA 2012 [2]) na Slovensku, kde byl přijat a publikován odborný článek Tool for Desktop Sharing and Remote Teaching - ForceB [3]. Konference je organizována pod záštitou organizace IEEE [4]. Dále se tento projekt účastnil 8. ročníku semináře ORGANON [5], kde byl také publikován odborný článek.

2 Webináře

Webináře [8] dnes začínají plnit velkou roli v e-learningových systémech. Jedná se o velmi propracovanou technologii se spoustou možností v přístupu k výuce. Webináře jsou určeny především ke školení a k výuce online. Jedná se tedy o systémy provozované v prostředí Internetu. Název vznikl ze spojení slov webový seminář. Historicky se toto pojmenování začalo používat od 21. století, kdy se rozšířily videokonference.

2.1 Charakteristika

Pro aplikace využívající principu webináře je charakteristická komunikace, která probíhá prostřednictvím internetu v prostředí virtuálních učeben. Výuka je vedena lektorem, který optimálně kombinuje přednosti osobního a e-learningového vzdělávání. Jedná se o alternativu k běžnému fyzickému školení, jež má převážně informativní charakter a nevyžaduje fyzickou přítomnost na konkrétním místě. Studenti mohou používat běžný hardware a software. Službu lze využívat pro online výuku nejrůznějších předmětů, školení na určitý produkt nebo službu.

2.2 Způsob komunikace

Komunikace probíhá oběma směry a umožňuje plné zapojení studentů. Ti mají možnost přímé interakce s přednášejícím pomoci chatu, hlasem nebo mohou kreslit a psát přímo na tabuli vyučujícího. Tyto funkce je možné omezit a zpřístupnit ve vhodnou chvíli.

V závislosti na použitém systému může přednášející využívat nejrůznější nástroje pro zvyšování interaktivity a prohloubení zážitků z webináře. Většina moderních systémů dovoluje s účastníky sdílet prezentace, obrázky, videa, webové stránky nebo pracovní plochu obrazovky. Mezi další funkce patří například ankety, které lze během semináře nechat studenty vyplnit. Díky tomu může mít vyučující okamžitou zpětnou vazbu o kvalitě výuky. Z každého semináře je také obvykle možné pořídit video záznam. Ten lze publikovat na veřejných serverech jako je například youtube.com nebo ho zpřístupnit pouze úzkému okruhu lidí. Přístup do virtuálních učeben může být také omezen pouze pro registrované účastníky nebo otevřen široké veřejnosti.

2.3 Možnosti využití webinářů

Možnosti využití webinářů jsou velké. Nemusí jít pouze o vzdělávání studentů a pracovníků z různých koutů světa, ale pomocí webinářů se můžou pořádat i marketingové akce pro získání nových zákazníků. Lze je využít k představení novinek a nových technologií, k pořádání online proslovů, k online konferencím nebo k setkání s odborníky a spoustu dalších. Všem zúčastněným stranám to navíc přináší značné úspory času a nákladů. Účastníci i pořadatelé se mohou snadno připojit přímo z kanceláře nebo domova a nemusí nikam cestovat. Tím získávají možnost účastnit se různých akcí, které by byly za normálních okolností nedostupné, například z důvodu velké vzdálenosti nebo časové náročnosti.

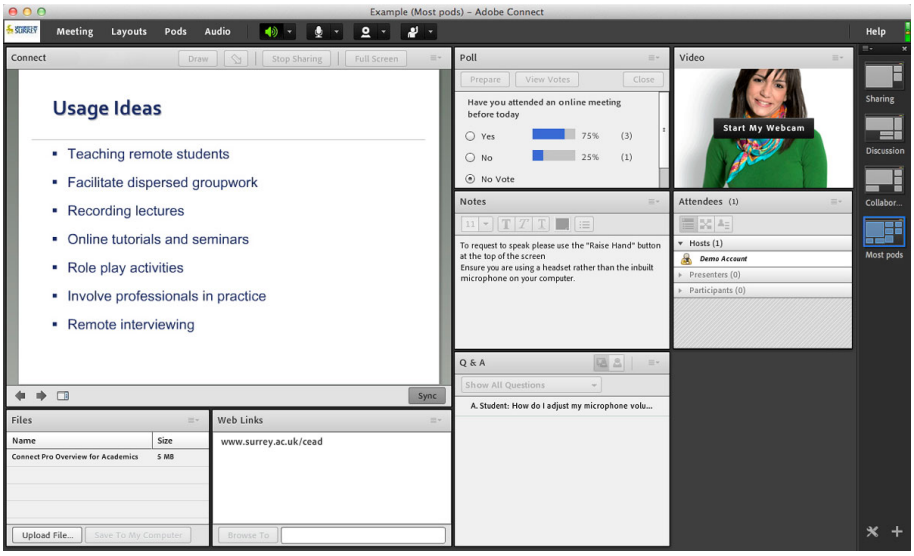
3 Existující systémy pro pořádání webinářů

Tato část je věnována současným systémům pro pořádání webových seminářů a jejich srovnání. V rámci své bakalářské práce jsem již podobnou analýzu realizoval. Díky tomu zde budou popsány rozdíly mezi staršími verzemi jednotlivých systémů a jejich zlepšení. Většina systému doznala jen estetických úprav, případně rozšířila podporu i na mobilní platformy. Není tedy důvod znovu vyjmenovávat všechny funkce a případné zájemce odkáži na svou bakalářskou práci.

Na poli webových seminářů se neobjevil žádný nový velký hráč. Jsou zde stále osvědčené firmy jako je Adobe se svým Acrobat Connect, dále systém WebEx od světoznámého výrobce síťových prvků, firmy Cisco a v neposlední řadě také systém GotoWebinar. Ten je výrobkem americké společnosti Citrix.

3.1 Adobe Acrobat Connect

Mezi lídry v oblasti webových seminářů patří stále řešení Acrobat Connect [16] od firmy Adobe. Jejich e-learningový systém pro pořádání webinářů a videokonferencí patří mezi řešení, které svou propracovaností a robustností uspokojí většinu uživatelských požadavků na podobné systémy. Zaměřen je především na střední a velké firmy. Umožňuje komunikovat a spolupracovat s až 2500 účastníky prostřednictvím snadno použitelných a lehce přístupných osobních konferenčních místností 1.



Obrázek 1: Ukázka prostředí služby Adobe Acrobat Connect Pro

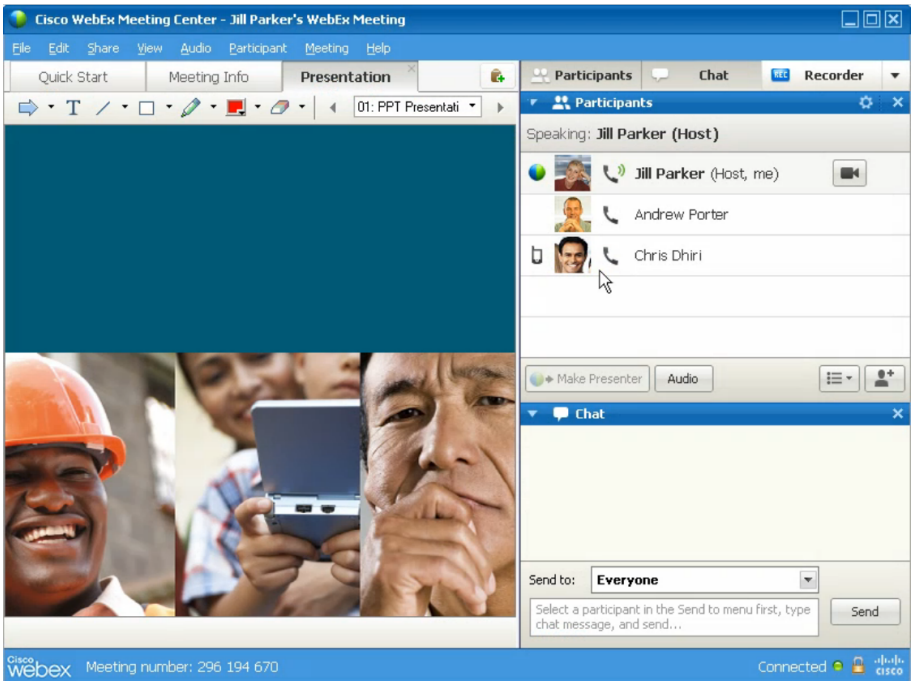
Prostředí systému Adobe Connect je intuitivní. Doznalo řady změn a vylepšení. Nyní působí daleko uceleněji a nabízí mnoho nových možností. Lze si například dynamicky volit rozložení jednotlivých prvků na obrazovce a přibylo mnoho nových funkcí. Nyní je možné rychle sdílet soubory, odkazy na webové stránky, poznámky a mnoho dalších.

Jednotlivé funkce jsou reprezentovaný malými boxy, které lze libovolně přemisťovat nebo skrývat. Ke každému boxu je možné nastavit oprávnění pro studenty.

Firma Adobe se ve vývoji zaměřila hlavně na mobilní segment a uvolnila aplikace pro většinu mobilních platforem. Zastoupení zde najdou systémy iOS, Android i BlackBerry. Aplikace jsou uvolněny jak pro mobilní zařízení, tak pro tablety. Tyto aplikace umožňují využívat všechny možnosti webových seminářů a studenti nebo obchodníci tak nejsou ničím omezováni.

3.2 Cisco WebEx

Společnost Cisco v tomto segmentu nabízí široký sortiment služeb a aplikací pod značkou WebEx [17]. V oblasti školení a online webinářů to jsou především služby Meeting center, Training center a Event center. Jednotlivé služby dokáží pokrýt potřeby malé firmy o pár zaměstnancích ale i velké společnosti s vysokými nároky na stabilitu a rychlost. Služby jsou poskytovány formou SaaS [13]. Firma pořádající webináře tak nemusí investovat prostředky do vlastního hardwaru a nové infrastruktury. Společnost Cisco již dříve vynikala svou obrovskou výpočetní a datovou kapacitou. Ani nyní se na tom nic nezměnilo, umožňuje realizovat i velké online semináře pro mnoho tisíc účastníků 2.



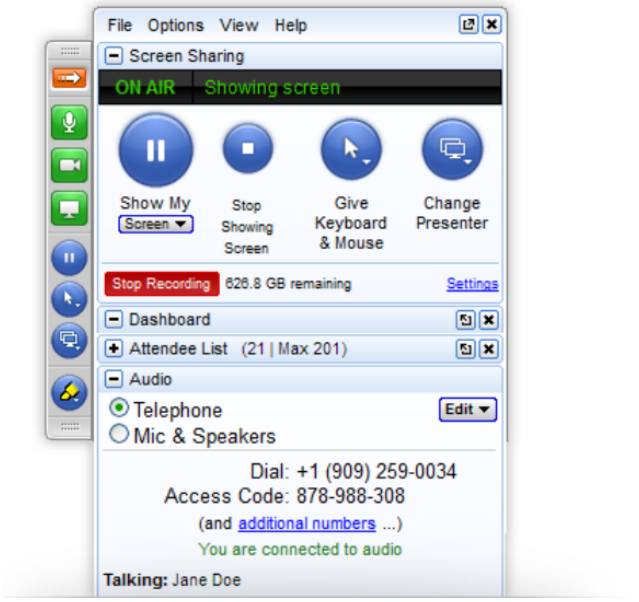
Obrázek 2: Ukázka prostředí služby Cisco WebEx

Podobně jako v systému od Adobe i Cisco dalo svým aplikacím nový grafický design. Rozložení jednotlivých prvků zůstalo stejné. Po pravé straně najdeme seznam přihlášených účastníků, chat a poznámky. Zbytek plochy můžeme vyplnit whiteboardem, pre-

zentací nebo sdílením multimediálních souborů. Systém umožňuje sdílet až 6 webkamer současně.

3.3 Citrix GoToWebinar

Firma Citrix se specializuje na software podporující kolaboraci a ovládání PC na dálku. Kromě služby GoToWebinar [18] vyvíjí i řešení GoToMeeting, GoToTraining a GoToAssist. Jak názvy jednotlivých služeb vypovídají, systém od společnosti Citrix umožňuje nejenom organizaci webinářů ale i schůzek, prezentací, školení a mnoho dalších. Každá ze služeb je zaměřena pro jinou cílovou skupinu. Jednotlivé služby dokáží pojmout řádově jednotky až tisíce účastníků. Jednotlivé služby se neliší pouze funkcemi ale především cenou 3.



Obrázek 3: Ukázka prostředí služby Citrix GoToWebinar

Prostředí pro pořádání webinářů nezaznamenalo praktický žádných změn. Nepříjemnou podmínkou používání této služby je nutnost instalace desktopové aplikace. Systém na rozdíl od jiných služeb neběží v okně webového prohlížeče. Ovládání aplikace je poměrně intuitivní a obsahuje jak klasický chat, tak možnost sdílet video a pracovní plochu.

4 Shrnutí bakalářské práce

V bakalářské práci jsem se věnoval vývoji nástroje pro sdílení pracovní plochy a vzdálenou výuku. Výsledkem práce je systém pro pořádání webových seminářů umožňující výuku v reálném čase pomocí internetu nebo univerzitní sítě. Komunikace zde probíhá ve virtuálních učebnách. Jedná se o online kolaborativní platformu, která může sloužit pro výuku nejrůznějších předmětů, školení na určitý produkt či službu.

Přednášející pro výuku může použit několik nástrojů. Systém umožňuje sdílet pracovní plochu obrazovky, sdílet okno konkrétní aplikace nebo sdílet prezentace vytvořené pomocí aplikace Microsoft PowerPoint [28]. Komunikace v tomto systému probíhá oběma směry a umožňuje plné zapojení studentů. Studenti mají možnost přímé interakce s přednášejícím pomocí chatu, hlasem nebo mohou kreslit a psát přímo na tabuli přednášejícího. Tyto funkce je možné omezit a zpřístupnit ve vhodnou chvíli.

V diplomové práci jsem navázal na tento projekt a dále ho rozvíjel. Největší slabinou systému byla komunikace, která byla navrhnuta pro hvězdicovou topologii sítě. V centru sítě byl jeden server, který přeposílal komunikaci mezi všemi klienty. Nevýhodou tohoto řešení bylo omezení jednoho serveru, který dokázal obsloužit jen omezený počet klientů. Proto byla pro komunikaci místo hvězdicové topologie sítě zvolena topologie stromová. S tím souvisel i vývoj nového komunikačního protokolu FBP 6, který by umožňoval v takto navržené síti komunikaci mezi všemi prvky. Bylo nutné navrhnout systém pro směrování datagramů v síti a vyřešit problémy spojené s přenosem dat pomocí UDP protokolu [34]. Dále se diplomová práce soustředí na vývoj nového desktopového klienta, který je nyní naimplementován pomocí technologie WPF (Windows Presentation Fundation) [29]. Od začátku byla navrhnuta a naimplementována i práce se zásuvnými moduly, které nyní umožňují daleko větší kontrolu nad samotným seminářem. Vylepšen byl i systém komprese dat, zejména pak čtvercová komprese. Dále byla přenesena databázová strana systému z technologie společnosti Oracle na databázi Microsoft SQL Server.

Celá serverová část systému byla také od počátku vyvíjena pro prostředí cloudu. Konkrétně pro Microsoft Azure, který umožňuje dynamicky navyšovat množství serverů, mezi které je možné zátěž rozložit. Díky tomu je nyní možné pořádat kurzy až pro stovky klientů.

5 Přenos dat v internetu

Webové semináře jsou velmi náročné na komunikační síť. Data je nutné přenášet v reálném čase. Z toho důvodu hraje velkou roli nejenom šířka pásma, tzv. bandwidth [30], ale i zpoždění a QOS [31]. Ať už je seminář provozován na univerzitní síti nebo internetu je potřeba mít dostatečně velkou propustnost sítě. Největším problémem není přenos dat od prezentátora na server, ale přenášet tyto data ze serveru ke zbývajícím klientům. Počet klientů může být velký a server musí zajistit odeslání dat všem klientům s co nejmenší latencí.

5.1 Metody odesílání IP datagramů

5.1.1 Multicast

Již nějakou dobu existují technologie, které řeší odesílání dat z jednoho zdroje skupině více koncových stanic jako je například multicast [15, 35] nebo broadcast [35]. Místo odesílání jednotlivých datagramů každému cíli, je odeslán jediný datagram. Ten se v případě vyžádání v jednotlivých uzlech sítě (většinou směrovačích) replikuje a dále se posílá jen na ta síťová rozhraní, za nimiž se nacházejí zaregistrovaní příjemci dané relace. Odpadá tak potřeba data odesílat každému příjemci zvlášť. Cílem je tedy zmenšení zátěže vysílajícího uzlu a zefektivnění distribuce dat v přenosové síti.

Problémem technologie přeposílání datagramů je ale v potřebě zajistit na všech prvcích v síti (routery, switche, atd.) správnou konfiguraci pro tento typ přenosu. Toho můžeme dosáhnout ve firemních nebo lokálních sítích ale nikoliv v internetu. Existují sítě soukromých poskytovatelů internetu jako je Telefonica nebo akademické sítě jako je CESNET2 [32], které multicastový přenos podporují. Pro provoz webového semináře v prostředí internetu je multicastový přenos dat nevhodný.

5.1.2 Unicast

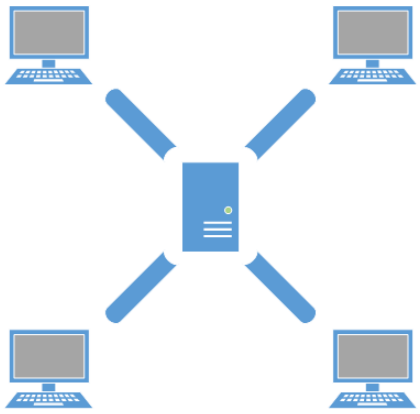
Unicast [14] je typ přenosu, kdy se data přenášejí pouze jedinému cíli. Tímto cílem může být uzel, nebo cílová stanice. Jedná se o nejčastěji používaný způsob přenosu dat v internetu. Unicast využívají protokoly jako je http, ftp, pop3, smtp, atd. U těchto protokolů spolu komunikují vždy pouze dvě stanice. A právě tento typ přenosu dat byl zvolen pro aplikaci ForceB.

5.2 Topologie sítě

Při pořádání webových seminářů potřebujeme přenášet totožná data od prezentátora ke všem připojeným klientům. V tomto případě nejsou obvykle data přenášena od prezentátora přímo k přihlášeným klientům. Prezentátor obvykle nedisponuje dostatečně velkou šířku pásma pro obsluhu všech připojených klientů. Pokud navíc počítáme s klienty komunikujícími v prostředí internetu, musel by prezentátor mít veřejnou IP adresu. Je tedy nutné zvolit vhodnou topologii sítě.

5.2.1 Hvězdicová topologie

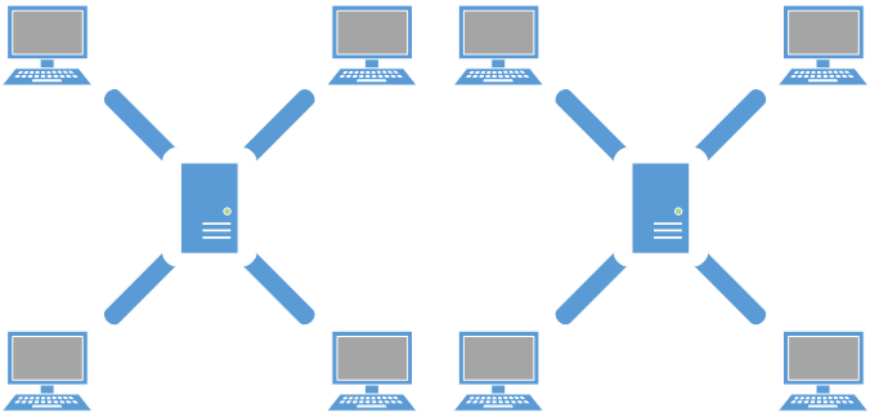
Pro komunikaci mezi více klienty se často volí topologie hvězdy [33], viz obrázek 4. Prezentátor data zašle na jeden centrální prvek, v tomto případě server, a ten je distribuuje všem přihlášeným klientům.



Obrázek 4: Hvězdicová topologie sítě

Tento způsob přenosu dat má jednu velkou nevýhodu. Pokud chceme stejná data zaslat více cílovým stanicím, každý paket se musí v případě unicastového přenosu zaslat každé stanici zvlášť. Server je z toho důvodu velmi limitován svou maximální šířkou pásma a může obsloužit jen omezený počet připojených klientů.

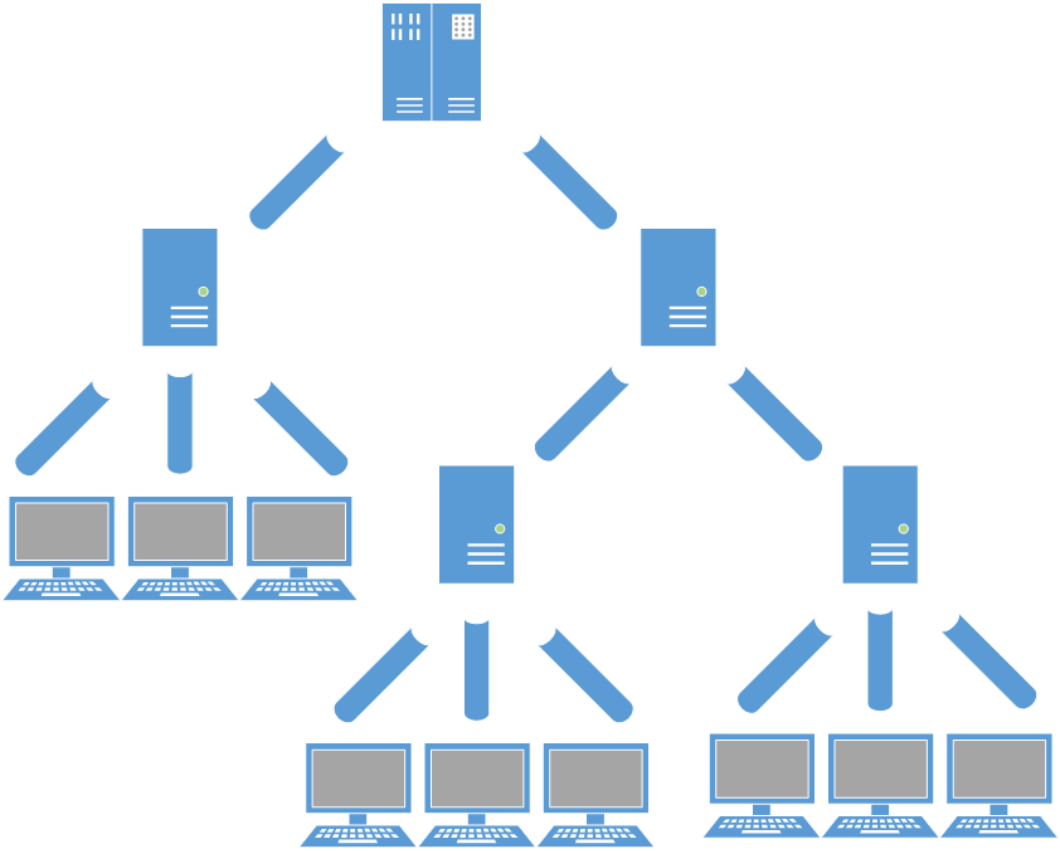
Tato topologie navíc neumožňuje zvolit možnost více serverů, mezi které by se zátěž rozložila. Typ topologie hvězdy nám to nedovolí. Založení dalšího serveru by znamenalo mít dva nezávislé okruhy, které spolu nejsou schopny komunikovat a předávat si potřebná data, viz obrázek 5. První server by byl schopen obsloužit jeden seminář a druhý server jiný seminář. Nebylo by tedy technicky možné pořádat semináře pro stovky nebo tisíce lidí.



Obrázek 5: Dvě nezávislé hvězdicové topologie sítě

5.2.2 Stromová topologie

Pro dynamické rozložení zátěže se nejlépe hodí topologie Tree [33] neboli stromu 6. Ta vychází z hvězdicové topologie spojením jednotlivých serverů, které jsou v centrech jednotlivých hvězd. Tuto topologii můžeme rozdělit do tří částí. První je centrální server, který tvoří kořen stromu (root). Tento server řídí samotnou komunikaci, data přes něj ale fyzicky nemusí proudit. Další důležitou části jsou uzly stromu, tzv. nody. Ty jsou připojeny k centrálnímu serveru nebo jsou propojeny mezi sebou. Právě ony slouží pro přeposílání komunikace mezi klienty. Každý z klientů je připojen nikoliv k centrálnímu serveru ale k jednomu z uzlů.



Obrázek 6: Stromová topologie sítě

V takto navržené síti musí probíhat směrování, na jehož základě můžeme přenášet data od libovolného klienta v síti k jinému. Toto směrování obstarává v rámci ForceB sítě vlastní FBP protokol. Podrobně se směrování v rámci ForceB sítě věnuje kapitola 6.4.

Díky této topologii můžeme navýšit maximální kapacitu na velikost, kterou bychom s jedním serverem nemohli dosáhnout. Pokud dojde k vyčerpání kapacity jednoho uzlu, každý další klient je připojen k uzlu jinému. Na základě směrování jsou poté data vždy dopravena ke svému cíli.

Hloubka stromu není nijak omezována. Limitován je pouze počet prvků připojených k jednomu uzlu. Ten je dán maximálním datovým tokem takového uzlu. Klienti by v rámci jedné relace neměli být rozčleněni mezi velké množství uzlů v různých hloubkách. Mohlo by to způsobovat zpoždění v přenosu dat a pomalé reakce na změny v síti zapříčiněné pomalou aktualizací směrovacích tabulek.

6 ForceB protokol – FBP

Pokud spolu chtějí dva koncové body komunikovat, musí existovat konvence, na jejíchž základě probíhá elektronická komunikace a přenos dat. Těmto konvencím nebo také standardu se říká protokol. Aplikace ForceB komunikuje na základě vlastního aplikačního protokolu FBP (ForceB protokol). Ten definuje pravidla, která se používají při vysílání a příjmu dat. Dále definuje řídící syntaxi, sémantiku a synchronizaci vzájemné komunikace.

Protokol obecně specifikuje mnoho vlastností, například:

- Vyjednávání o různých parametrech spojení.
- Jak začít a ukončit zprávu.
- Jak formátovat zprávy.
- Jak ukončit relaci nebo spojení.
- Co dělat s poškozenými nebo nesprávně naformátovanými daty.
- Jak detekovat neočekávanou ztrátu spojení a co provést jako další akci.

FBP protokol zajišťuje přenos dat mezi koncovými body v reálném čase. Tento protokol není závislý na transportním protokolu. Pro různé typy přenášených dat se doporučuje používat konkrétní transportní protokol. Pro multimediální data se nejčastěji používá UDP [34] protokol a pro textové data se využívá TCP [34] protokol. Při přenosu se používá pouze unicastový typ spojení. Multicast není podporován. ForceB protokol vychází z RTP (Real-time transport protocol) [6, 7]. Ten je internetovým standardem pro přenos dat v reálném čase. FBP ovšem není zpětně kompatibilní s RTP. Přejímá pouze některé jeho principy pro zpracování multimediálních dat.

FBP lze využít ve dvou typech topologii sítě. První je hvězdicová topologie. Jedná se o síť, kde je jeden centrální prvek, ke kterému jsou připojení ostatní klienti. Pokud spolu chtějí dva a více klientů komunikovat, musí jeden z klientů data nejprve zaslat na server a ten je přepošle cílovým klientům. Jak již bylo popsáno v předchozí kapitole, nevýhodou této topologie je její nemožnost rozložit zátěž mezi více serverů. Daleko vhodnější je proto použít stromovou topologií sítě, kde lze zátěž rozložit mezi několik uzlů.

Tento protokol byl navrhnut pro účely webového semináře. Je tedy zaměřen na komunikaci mezi prezentátorem a skupinou více koncových klientů. Zároveň umožňuje mnoha klientům, komunikovat se skupinou jiných klientů. Například v případě videokonferencí. Již ze samotného principu tedy nejde pouze o komunikaci jednoho k jednomu.

6.1 Architektura FBP

Přenos dat je v FBP rozdělen do dvou částí. První část tvoří režijní data. Zde spadají data týkající se jednotlivých účastníků. Jako jsou směrovací data, data o vysílaných proudech a statistická data. Obecně jsou tyto data nazývaná FBP Control Data 6.3. Druhou část přenášených dat tvoří již samotný obsah. Ten můžeme dále rozdělit na audio data, video data, textová data a mnoho dalších.

Celá komunikace mezi dvěma koncovými body je jednoznačně určena síťovou adresou a portem. Pro komunikaci přes UDP se používá port 23589 a pro TCP komunikaci port 23689. Pokud to konfigurace sítě dovoluje, tak jsou pro různé typy dat navázána další spojení na jiných portech. V případě použití UDP protokolu vzniká mezi dvěma účastníky při zahájení komunikace relace. Tuto relaci si můžeme představit obdobně jako u TCP/IP spojení. Je vytvořen plně duplexní kanál mezi dvěma koncovými body. Jsou zde také navrhnuty mechanismy pro potvrzování přenášených dat. Oproti TCP/IP spojení zde chybí mechanismus pro přeposílaní ztracených datagramů, tzv. retransmise. Tento proces je nežádoucí pro aplikace komunikující v reálném čase. Není potřeba ztracená data znovu přeposílat, protože jsou již zastaralá.

Pro všechna přenášená dat existuje vlastní spoj. Tomu je přidělen jedinečný identifikátor. Každý odeslaný datagram se poté identifikuje na základě tohoto identifikátoru. Tento spoj si můžeme představit jako proud dat. Může jít například o obraz z konkrétní webkamery nebo o zvuk z konkrétní zvukové karty. Díky identifikátoru v hlavičce každého zaslaného datagramu je možné jednoznačně odlišit jednotlivé proudy dat. Účastník si tak může zvolit pouze ten proud dat, o který má zájem. V případě nedostatečného připojení k internetu si může ve videokonferenci zvolit například přijímat pouze zvukovou stopu.

Pokud je pro přenos dat na transportní vrstvě využíván UDP protokol, tak není zaručeno, že jednotlivé datagramy budou doručeny. Může dojít k jejich ztrátě v průběhu přenosu. Datagramy mohou být také doručeny v různém pořadí. S těmito situacemi si musí umět poradit konkrétní implementace ForceB protokolu. Pro tyto účely jsou v hlavičce každého zaslaného datagramu obsaženy informace, nutné pro rekonstrukci odeslaných dat nebo pro detekci ztráty jednotlivých datagramů.

Hlavička každého odeslaného datagramu obsahuje informace o pořadí datagramu (sequence number). Díky tomuto příznaku je možné snadno odhalit, zda došlo ke ztrátě nebo k přehození některých datagramů a původní zprávu rekonstruovat. S pomocí dalších příznaků je možné zjistit, o jaký typ dat se jedná nebo jednotlivé proudy dat synchronizovat. Tyto a další příznaky v FBP paketu jsou podrobně popsány v následující kapitole.

6.2 Struktura záhlaví FBP paketu

Přesnou strukturu paketu je možné vidět v tabulce 1.

2b	1b	7b	4b	1b	1b	16b
V	**M**	**Payload type**	**EHS**	**C**	**E**	**Sequence number**
32b						
Timestamp						
32b						
SSRC						
32b						
Payload data						

Tabulka 1: Struktura záhlaví FBP paketu

Vysvětlení jednotlivých příznaků v záhlaví FBP paketu:

- **Verze (V) 2bit** - Určuje aktuální verzi aplikačního protokolu FBP. Aktuálně se jedná o verzi 1.
- **Marker (M) 1bit** - Tento bit je určen k označení některých částí ve streamu. Může například označovat poslední paket ve framu.
- **Payload type 7bit** - Identifikuje formát přenášených dat. Jednotlivé hodnoty jsou přesně definovány a nelze je zaměňovat.
- **Exctension header size (EHS) 4bit** - Velikost o jakou je rozšířené záhlaví datagramu. Slouží pro rozšíření záhlaví pro dodatečné informace.
- **Compression 1bit** - Určuje, zda jsou data v tomto datagramu komprimovaná.
- **Encrypting 1bit** - Říká, jestli jsou data v datagramu zašifrována.
- **Sequence number 16bit** - Sekvenční číslo, které je inkrementováno s každým odeslaným paketem. Příjemce dat za pomoci tohoto čísla dokáže rekonstruovat pakety do původního framu nebo detekovat jejich ztrátu. Toto číslo je voleno pro první paket z framu náhodně.
- **Timestamp 32bit** - Neboli časové razítko. Označuje časový okamžik odebrání prvního vzorku bytu užitečného obsahu. Určuje vzorkování a je závislý na přenášených datech a jejich vzorkovací frekvenci. Slouží k synchronizaci uvnitř média. Například k synchronizaci několika souběžných proudů dat jako je obraz a zvuk.
- **SSRC 32bit** - Jednoznačně identifikuje zdroj synchronizace FBP paketů.

6.2.1 Typy FBP paketů

Každý zaslaný paket je identifikován jednoznačným identifikátorem SSRC a hodnotou PayloadType. Ta určuje typ dat, která jsou obsažena v užitečném obsahu. Již na úrovni paketu lze říct, o jaká data se jedná a jakým způsobem budou dále zpracovávaná. Pokud to konfigurace sítě dovoluje, měl by pro každý typ paketu existovat nový spoj na samostatném portu. Pokud jsou porty blokovány, může celá komunikace probíhat na výchozím portu 23589.

Název paketů, spolu s číselnou hodnotou a portem je uveden v tomto seznamu:

- **FbpControlData (1) port 23589** – Tělo paketu obsahuje řídicí data důležitá pro směrování, statistiky, řízení přenosu a mnoho dalších. Podrobněji v následující kapitole.
- **FbpParticipant (2) port 23590** – Jedná se o data pro autentizaci a autorizaci klienta.
- **Chat (50) port 23591** – Textová data pro zasílání jednoduchých zpráv.
- **GroupChat (51) port 23591** – Textová data distribuovaná skupině příjemců.
- **FileTransfer (52) port 23592** – Užitečný obsah nese pole bytu souboru.
- **SqueareCompression (100) port 23593** – Video reprezentované snímky ve formátu jpeg s použitím čtvercové komprese.
- **MJPEG (101) port 23594** – Video reprezentované snímky ve formátu jpeg.
- **H.263 (102) port 23595** – Video H.263.
- **H.264 (103) port 23596** – Video H.264.
- **G.722 (104) port 23597** – Audio G.722.
- **G.728 (105) port 23598** – Audio G.728.
- **OtherData (106) port 23599** – Tento typ proudu dat slouží pro libovolná data. Dále si je specifikuje každá aplikace implementující ForceB protokol.

6.3 FBP Control Data pakety – FBPCD

V rámci komunikace mezi serverem, uzly a klienty probíhá neustálý přenos řídících dat. Pro přenos těchto dat je upřednostňován UDP protokol. Pokud by nebylo možné zvolit tento protokol, mohou být data přenášena i pomoci jiného protokolu transportní vrstvy.

Při přenosu jsou řídící data vkládána do FBP paketu, kde je PayloadType nastaven na FbpControlData. První byte užitečného obsahu paketu pak označuje typ řídících dat. Protokol zajišťuje multiplexování dat. Díky tomu mohou být FBPCD data zasílána na stejném portu 23589 spolu s užitečným obsahem. Doporučuje se však pro každý typ dat zvolit jiný port, jednotlivá čísla portů jsou definována v předchozí kapitole.

Mezi primární funkce řídících dat patří zajištění směrování v celé síti. ForceB protokol neomezuje hloubku stromu, a proto je nutné zajistit správnost a aktuálnost informací v každém uzlu. Toho se dosahuje díky pravidelnému zasílání směrovacích informací mezi jednotlivými uzly, klienty a serverem. V rámci této komunikace se nezasílají informace o všech účastnících všem účastníkům. Naproti tomu je navrhnut inteligentní systém, na jehož základě jsou distribuována pouze potřebná směrovací data do každé části stromu. Podrobně se směrování ve ForceB protokolu věnuje kapitola 6.4.

Řídící data můžeme rozdělit do dvou typů podle toho, kdy jsou odesílána. Prvním typem jsou data zasílaná v pravidelných intervalech. Jedná se především o směrovací data a o data spojená se statistikami přenosu. Snahou je zvolit takový časový interval, který je dostatečně velký pro rychlou aktualizaci dat a zároveň nezahlcuje datovou linku víc, než je nutné. Tato signalizace je zasílaná vždy v obou směrech komunikace. Jak od vysílací strany k přijímající tak opačně. V kapitole 6.4.2 budou vysvětlena přesná pravidla přenosu těchto informací.

Druhým typem řídících dat jsou data zasílaná na určitý podnět. Zde patří především data zajišťující řízení datového toku nad UDP protokolem. Na jejich základě lze adaptivně upravovat rychlost, s níž jsou data odesílána, dále potvrzovat inicializaci streamu nebo odhlašovat účastníky z relace. Podle potřeby jsou data zasílána jak od přijímající strany k vysílací, tak opačně.

6.3.1 Architektura FBPCD

Formát v jakém jsou FBP Control Data serializována do pole bytu je přesně definován a je dán typem řídících dat. Jedná se o strukturu, kde první byte určuje typ FBPCD a za ním následují v řadě jednotlivé property. Pořadí a typ jednotlivých property je definován pro každý typ řídících dat. Tyto property vždy obsahují typ dat, jejich velikost a užitečný obsah. Schéma lze vidět v tabulce níže.

Type FBPCD	**Property1**			**Property2**		
8bit	8bit	16bit	Length bit	8bit	16bit	Length bit
Type BPCD	TypeCode	Length	PayloadData	TypeCode	Length	PayloadData

Tabulka 2: Struktura FBPCD paketu

Popis jednotlivých příznaků:

- **Type FBPCD (8bit)** – Typ FBP Control dat, viz číselník v následující kapitole.
- **TypeCode (8bit)** – Určuje datový typ užitečného obsahu. Jednotlivé hodnoty jsou následující:

  Empty = 0, Object = 1, DBNull = 2, Boolean = 3, Char = 4, SByte = 5, Byte = 6, Int16 = 7, UInt16 = 8, Int32 = 9, UInt32 = 10, Int64 = 11, UInt64 = 12, Single = 13, Double = 14, Decimal = 15, DateTime = 16, String = 17

- **Length (8bit)** – Velikost užitečného obsahu v bytech.
- **PayloadData (Length bit)** – Má proměnlivou délku. Uchovává samotný obsah property.

6.3.2 Typy FBPCD paketů

V následující kapitole budou popsány jednotlivé typy FBP Control Dat. Bude zde vždy uvedena struktura přenášených dat. Dále zde budou popsány funkce jednotlivých paketů a intervaly zasílání.

6.3.2.1 FbpRoutingData: Pakety s těmito daty obsahují směrovací informace. Ty slouží pro určení všech cest v síti a jako celek vytváří mapu, za pomoci které je možné pakety s užitečným obsahem doručit ke svému cíli. Tato data jsou generována každým prvkem v rámci ForceB protokolu. Může se jednat o koncovou stanici, uzel nebo server.

Informace v jednom paketu se vždy vztahují k jednomu subjektu a k jeho proudům dat. Každý prvek si na základě přijatých paketů od okolních prvků vytváří vlastní směrovací tabulku. S každým přijatým paketem dochází k její aktualizaci a v případě změny, je tato změna distribuována okolním prvkům.

Generování dat probíhá před každým odesláním. Navíc jsou data generována zvlášť pro každý prvek, se kterým je daný účastník ve spojení. To znamená, že pro každý směr komunikace jsou odesílána odlišná data. Odesílání probíhá vždy v pravidelných intervalech. Pokud ale nastane na některém z prvků změna ve směrovací tabulce (může jít například o vytvoření nového proudu dat nebo o přidání nového cílového klienta k existujícímu proudu), tak je tento prvek povinen tuto změnu v co nejkratším čase distribuovat všem okolním prvkům.

FbpRoutingData pakety obsahují tyto property (hodnoty v závorkách udávají jejich typ):

- **SSRC (uint)** – Jedinečný identifikátor subjektu, kterého se tato data týkají. Ne vždy se tyto informace vztahují k subjektu, od kterého data přišla.
- **ConnectedType (Enum)** – Určuje v jakém vztahu je daný prvek k přijímající straně. Může nabývat hodnot: DirectlyConnected, ServerDirectlyConnected, ConnectedThroughNode.
- **StreamDataList (FbpStreamData[])** – Obsahuje pole s FbpStreamData. Ta nesou informaci o inicializovaných proudech dat na straně vysílajícího prvku, viz následující kapitola. Tento seznam je distribuován pouze bezprostředně připojeným prvkům. Pokud se jedná o stromovou topologii sítě, tak se dále distribuuje pouze jedinečný identifikátor a jako ConnectedType je nastaven ConnectedThroughNode.

6.3.2.2 FbpStreamData: V této struktuře jsou obsaženy informace o jednom vysílaném proudu dat. Informace o všech vysílaných proudech dat jednoho prvku jsou poté uloženy jako pole v StreamDataListu. Toto pole je obsaženo v FbpRoutingData paketech a zasílané jsou společně.

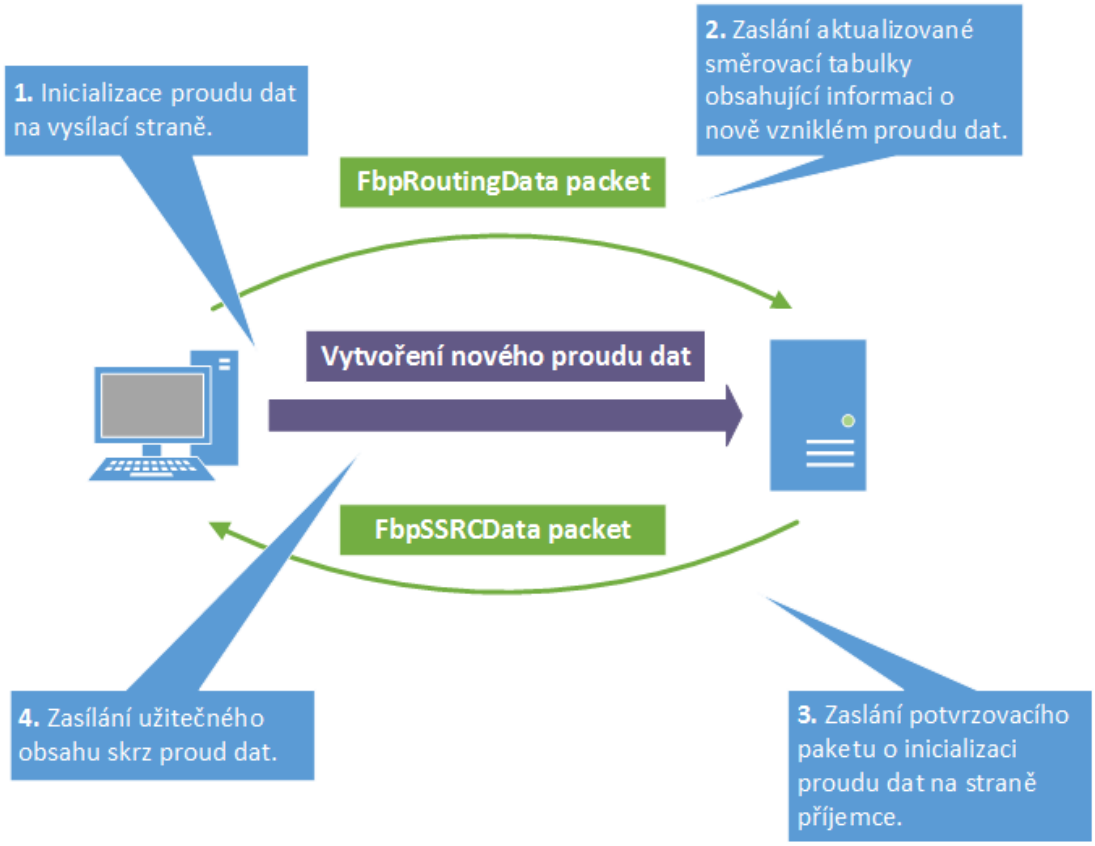
Informace v FbpStreamData paketech jsou důležité pro inicializaci proudu dat na straně příjemce. Proto je důležité, aby vysílající strana po inicializaci nového proudu dat v co možná nejkratším čase zaslala aktualizovanou směrovací tabulku směrem k přijímající straně. Pokud by některému prvku v síti začaly chodit datagramy s užitečným obsahem a daný prvek by ve své směrovací tabulce neměl o těchto datech žádné informace, tak je povinen tyto datagramy zahazovat.

FbpStreamData obsahují tyto property:

- **SenderSSRC (uint)** – Udává jedinečný identifikátor vysílaného proudu dat. Každý paket užitečného obsahu poté v hlavičce nese tento identifikátor.
- **PayloadType (Enum)** – Typ přenášených dat v užitečném obsahu daného proudu dat. Konkrétní číselník možných hodnot je uveden v kapitole 6.2.1. Tato hodnota může být v průběhu přenosu změněna. Například, když se přejde na vyšší či nižší úroveň komprese.
- **InternalName (string)** – Jde o interní název zařízení, které generuje data pro tento proud dat. Může jít například o jméno web kamery nebo o jméno plug-inu. Používá se pro inicializaci potřebných zařízení a knihoven na straně příjemce.
- **TargetSSRC (uint[])** – Obsahuje pole cílových zařízení, reprezentované jedinečnými identifikátory těchto zařízení v rámci celé sítě ForceB. Na základě těchto hodnot je každý uzel schopen data v rámci sítě dále směrovat. Pokud klient tuto hodnotu nenastaví, znamená to, že data mají být doručena serveru. Toho se využívá především při inicializování spojení nebo při dotazech na server. Například pro získání podrobnějších informací o kurzu a podobně.
- **SourceSSRC (uint)** – Nese informaci o klientovi, který inicializoval proud dat. Pokud jsou data přeposílána přes několik prvků, tak cílový prvek vždy ví, od kterého zdroje data pochází. V případě videokonference, které se účastní několik účastníků, je každý z klientů
- **Confirmed (bool)** – Tato hodnota udává, zda je daný proud dat inicializován na straně příjemce. Při inicializaci se na vysílací straně nastaví na hodnotu false.

6.3.2.3 FbpSSRCData: Informace obsažené v těchto paketech slouží ke dvěma účelům. Každý z prvků v síti je povinen při ukončení relace zaslat paket s informací o ukončení relace. Tento paket obsahuje jedinečný identifikátor daného účastníka, tzv. SSRC a jako ActionType má nastavenou hodnotu Bey. Tato informace je poté dále distribuována mezi všemi uzly až k serveru. Všechny prvky, které dostanou tuto informaci, regulérně odhlásí daného účastníka a následně po něm uvolní všechny alokované prostředky.

Další funkcí těchto paketů je potvrzení inicializace proudu dat na straně příjemce. Ve chvíli kdy vysílací strana inicializuje proud dat, začne odesílat k příjemci aktualizovanou směrovací tabulku. Ta mimo jiné obsahuje informaci o nově inicializovaném proudu dat. Zde je hodnota Confirmed nastavena na false. Na základě těchto informací přijímající strana inicializuje proud dat na své straně. Poté je směrem k vysílací straně zaslán FbpSSRCData paket, který potvrzuje inicializaci dat na straně příjemce a vysílací strana může v tuto chvíli začít odesílat užitečný obsah. Tento proces znázorňuje obrázek 7.



Obrázek 7: Proces inicializace proudu dat

Pokud by došlo během přenosu přes UDP protokol ke ztrátě potvrzovacího FbpSSRCData paketu, tak vysílací strana bude v rámci pravidelného zasílání směrovací tabulky stále informovat o nepotvrzeném proudu dat pomocí hodnoty Confirmed. Přijímající strana po obdržení této hodnoty opět zašle potvrzovací FbpSSRCData paket. Tím je zaručeno, že dřív nebo později k inicializaci proudu dat na straně příjemce dojde.

FbpSSRCData obsahují tyto property:

- **SSRC (uint)** – Může obsahovat číslo identifikující konkrétní proud dat nebo konkrétního účastníka přenosu. V závislosti na nastavené hodnotě ActionType.
- **ActionType (Enum)** – Tato property může nabývat dvou hodnot: Bye, Confirm. Ty určují, zda má být účastník nebo proud dat ukončen nebo jde o data potvrzující inicializaci proudu dat na straně příjemce.

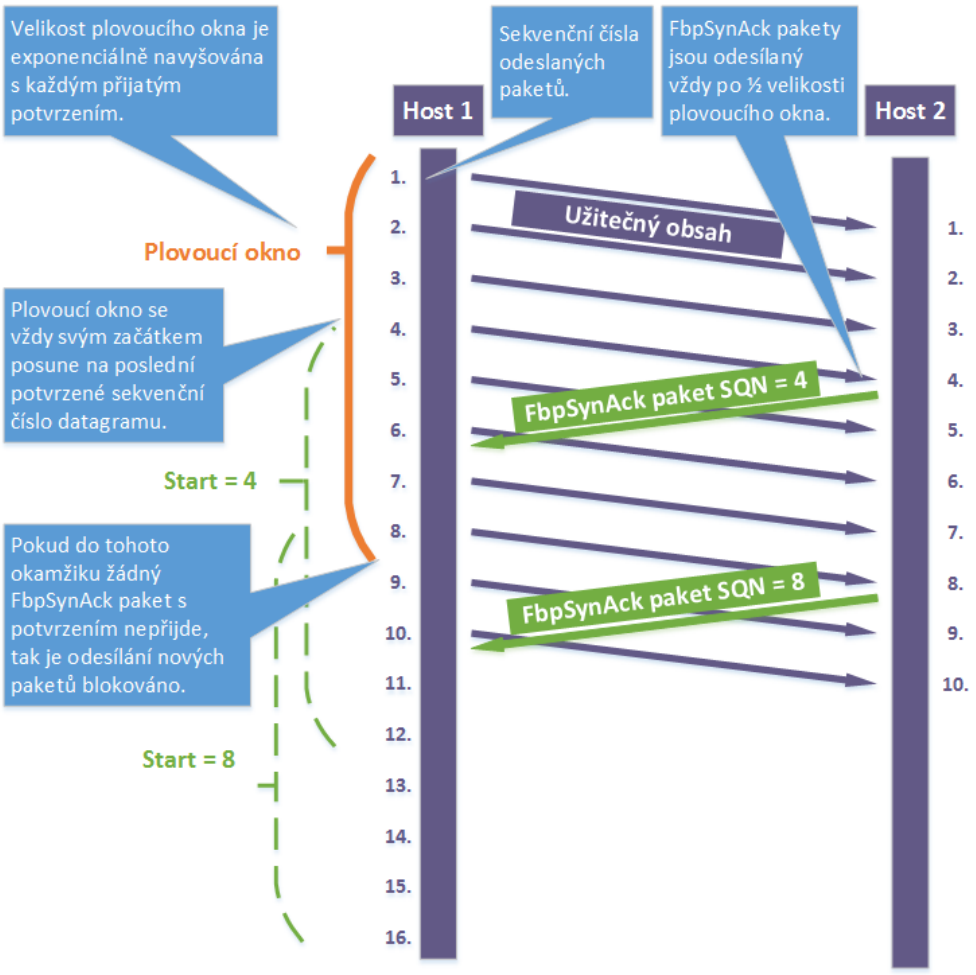
6.3.2.4 FbpSynAckData: Pakety s touto informací slouží pro řízení datového toku nad UDP protokolem. TCP protokol vytváří pro aplikace určitou abstrakci souvislého datového proudu. Aplikace pouze zasílá svá data do virtuální roury, na jejímž konci vyjdou nepoškozená data v tom pořadí, jak byla poslána. Aplikace nemusí řešit situace, kdy se po cestě ztratí některý z paketů nebo že je hrdlo mezi serverem a klientem příliš úzké. Všechny tyto problémy vyřeší na pozadí TCP protokol. Toto ale neplatí v případě UDP protokolu. ForceB protokol, který využívá UDP, se s úzkým hrdlem mezi serverem a klientem vypořádává vlastními metodami.

Pokud hovoříme o komunikaci v internetu, tak obvykle nastává situace, kdy je mezi dvěma koncovými body různá velikost šířky přenosového pásma. Jestliže necháme v takovém případě vysílací stranu odesílat datagramy s rychlostí, jakou je generuje, začnou tyto datagramy plnit zásobník na některém síťovém rozhraní v síti. Po přetečení takového zásobníku se nově přijaté datagramy začnou zahazovat a nikdy již nedorazí ke svému cíli. Proto je důležité rychlost přenosu dat určitými mechanizmy upravovat a neodesílat větší množství dat, než je koncový bod schopen přijímat.

ForceB protokol řeší tuto situaci průběžným potvrzováním již přijatých datagramů. Právě k tomu slouží FbpSynAckData pakety. Celý proces funguje tak, že se odešle pouze určité množství datagramů, z počátku velmi malé, a celý proces odesílání se pozastaví. Pozastaven je jen do chvíle, než přijde potvrzovací paket s číslem posledního přijatého datagramu na straně příjemce. Množství odeslaných datagramů závisí na velikosti klouzacího okna. Klouzací se mu říká proto, že se postupně posunuje s každým potvrzeným datagramem. Pro lepší znázornění je přiložen obrázek 8.

Z obrázku je patrné, že k odesílání potvrzovacích paketů nedochází po každém přijatém datagramu. Pokud by se potvrzoval každý přijatý datagram, neúměrně by to zatížilo linku směrem k odesilateli užitečného obsahu a celý proces by se zpomalil. Potvrzovací pakety se proto zasílají v určitých intervalech na základě sekvenčního čísla a velikosti klouzacího okna. Zasílají se vždy minimálně po 1/2 velikosti klouzacího okna. Pokud velikost okna nastavíme na 8, potom se potvrzovací paket zašle po každém čtvrtém sekvenčním čísle datagramu.

Velikost klouzacího okna udává množství paketů, které můžou být najednou odeslány. Pokud je sekvenční číslo odesílaného paketu větší, než je velikost klouzacího okna navýšená o sekvenční číslo posledního potvrzeného paketu, potom je odesílání pozastaveno. S každým přijatým potvrzením je posunut počátek klouzacího okna na poslední potvrzené sekvenční číslo.



Obrázek 8: Proces potvrzování doručených datagramů

Úmyslně se potvrzení zasílá vícekrát, než je velikost klouzacího okna. Musíme totiž počítat s určitým zpožděním mezi odesláním datagramu a přijetím potvrzení. Pokud by se potvrzení odeslalo pouze jednou a to na konci klouzacího okna, došlo by k pozastavení na straně odesilatele užitečného obsahu. Toto pozastavení by bylo na dobu než by poslední datagram přišel k příjemci a ten zpracoval a odeslal potvrzovací paket směrem k odesilateli užitečného obsahu. Teprve po přijetí potvrzovacího paketu by došlo u odesilatele k posunutí začátku klouzacího okna na potvrzené sekvenční číslo. Navíc může v případě UDP protokolu dojít ke ztrátě tohoto potvrzení. Z těchto důvodů se odesílá v jednom intervalu velikosti klouzacího okna více těchto potvrzení. Minimálně po 1/2 velikosti klouzacího okna. Optimální hodnota zjištěna měřením je po 1/4 velikosti klouzacího okna.

Jednou z nevýhod tohoto potvrzování je vysoká režie přenášených FbpSynAck paketů. Proto je nutné množství těchto paketů minimalizovat. Toho je docíleno postupným

zvětšováním klouzacího okna. Z počátku je velikost klouzacího okna nastavena na velmi malou hodnotu. S každým přijatým potvrzením se exponenciálně zvětšuje. Velikost okna se dále upravuje na základě statistických dat, kde se bere v úvahu aktuální přenosová rychlost a množství ztracených paketů za poslední periodu. Pokud dochází ke ztrátám paketů, okno je zmenšováno na optimální hodnotu.

FbpSynAckData obsahují tyto property:

- **SSRC (uint)** – Jedinečný identifikátor proudu dat, kterého se toto potvrzení týká. I když komunikace běží na jednom portu, tak jsou potvrzení zasílaná pro každý proud dat zvlášť. Díky tomu můžeme nastavit prioritu pro přístup k lince a zvýhodnit tak třeba zvuk před obrazem.
- **AcknowledgementNumber (ushort)** – Udává sekvenční číslo posledního přijatého datagramu.
- **Widnows size (ushort)** – Pomocí této hodnoty říkáme odesílací straně, jakou má nastavit velikost klouzacího okna. Tato hodnota se dynamický mění na základě několika faktorů. Mezi ně patří aktuální rychlost přenosu a množství ztracených paketů.

6.3.2.5 FbpPerformanceCounters: Tyto pakety slouží jako kontejner pro statistické informace všech vysílaných proudů dat. Pro tyto účely je zde pole FbpSenderStreamPC. Jednotlivé položky v tomto poli poté reprezentují statistické informace všech proudů dat. Odesílány jsou tedy všechny statistické údaje najednou. Nedochází tak ke zbytečné režii spojené se serializací a paketizací každého záznamu zvlášť.

Statistická data jsou odesílána v pravidelných intervalech. Tento interval je libovolný. Záleží na datových možnostech každého účastníka. Pokud je šířka přenosového pásma nízká, může docházet ke zvětšování intervalu mezi odesílanými pakety. Tyto pakety nejsou stěžejní částí protokolu. Slouží pouze jako podpůrný prostředek pro zjišťování stavu linky a případně zaznamenávání historie těchto dat do databáze.

FbpPerformanceCounters obsahuje tyto property:

- **FbpSenderStreamPCList** – Hodnota obsahuje pole FbpSenderStreamPC.

6.3.2.6 FbpSenderStreamPC: V této struktuře se ukládají aktuální statistické údaje konkrétního proudu dat. Tyto informace jsou poté zasílány protější straně. Ta na jejich základě může přizpůsobovat velikost klouzacího okna nebo tyto údaje dále přeposílat směrem k serveru. Pokud se jedná o statistické údaje, o vytížení jednotlivých uzlů, tak jsou tyto informace dále zpracovávány. V případě přetížení uzlů se může na jejich základě inicializovat nový uzel, který tuto zátěž dále rozloží.

Tato struktura uchovává statistická data jak na straně odesilatele daného proudu dat (sender), tak na straně příjemce proudu dat (stream). Na přijímající straně se navíc eviduje celkový počet ztracených paketů a rámců.

FbpSenderStreamPC obsahuje tyto property:

- **SSRC (uint)** – Jedinečný identifikátor proudu dat, kterého se tyto statistiky týkají.
- **Bytes (uint)** – Celkový počet bytů, které protekly přes tento proud dat.
- **Packets (uint)** – Celkový počet paketů.
- **PacketsLost (uint)** – Celkový počet ztracených paketů.
- **FrameLost (uint)** – Celkový počet ztracených rámců.
- **BytesPerSecond (uint)** – Průměrná přenosová rychlost v bytech za sekundu.
- **PacketsPerSecond (uint)** – Průměrný počet paketů za sekundu.
- **FramesPerSecond (uint)** – Průměrný počet rámců za sekundu.

6.4 Směrování ve FBP protokolu

Přidanou hodnotou ForceB protokolu je jeho schopnost směrovat data. To pomáhá k určování cest datagramů v prostředí sítě ForceB. Směrováním se zabývají všechny prvky v síti. Zajišťují ho nejen uzly a server, ale i koncové stanice. Jeho úkolem je zajistit doručení datagramů k adresátovi.

Směrování ve ForceB síti se trochu podobá směrování v počítačových sítích. Zde ovšem probíhá směrování dat mezi jednotlivými prvky na aplikační vrstvě ISO/OSI modelu. Datový tok je tedy řízen mezi jednotlivými prvky, které se účastní komunikace pomocí tohoto protokolu. Přenos paketu mezi jednotlivými prvky v počítačové síti je řízen nižšími vrstvami ISO/OSI modelu.

Obdobně jako v počítačových sítích, může být síťová infrastruktura sítě ForceB mezi odesilatelem a adresátem datagramu velmi složitá. Proto se směrování zpravidla nezabývá celou cestou datagramu, ale řeší vždy jen jeden krok, tj. komu datagram předat jako dalšímu.

Každý prvek v síti má sadu pravidel, která říkají, kam se mají předávat datagramy směřující k určeným cílům. Tato pravidla jsou obsažena v tzv. směrovací tabulce. Na jejich základě jsou data předána některému ze sousedních prvků a tam se rozhodovací proces opakuje podle zdejších pravidel. Není možné v každém uzlu sítě uvádět pravidla pro dosažení všech cílů. Jednak by to mohlo způsobovat kapacitní problémy v případě velkého množství účastníků a jednak by to zvyšovalo režijní náklady spojené s přenosem těchto směrovacích informací. Proto jsou pravidla při směrování zobecňována.

Samotné směrování je velmi odlišné od toho, jaké se používá v počítačových sítích. Zde jsou data směrována na základě jedinečného identifikátoru SSRC každého zaslaného datagramu. Nikoliv na základě cílového adresáta. A to z toho důvodu, že cílových adresátů může být libovolný počet. Přenášet tuto informaci v každém zaslaném datagramu nemá smysl. Proto v jednotlivých datagramech není žádný cílový adresát uveden. Tuto informaci nesou právě směrovací pakety.

6.4.1 Směrovací tabulka

Směrovací tabulka (anglicky routing table) obsahuje informace, které jsou nutné při rozhodování o dalším směřování paketu. Každý z prvků v sítí si vytváří vlastní směrovací tabulku. Změny v této tabulce, které souvisí s okolními prvky, jsou dále distribuovány. Neodesílají se tedy všechny informace ale pouze ty záznamy, které mají pro cílový prvek smysl.

Záznamy jsou ve směrovací tabulce rozděleny do řádků, přičemž každý obsahuje jednu směrovací informaci. Každý záznam se vztahuje k některému účastníkovi a obsahuje informace o jeho vysílaném proudu dat. Takový záznam mimo jiné obsahuje jedinečný identifikátor účastníka, identifikátor proudu dat, dále typ přenášených dat, seznam cílových adresátů a další. Všechny hodnoty obsažené ve směrovací tabulce včetně jejich podrobného popisu jsou uvedeny v kapitole 6.3.2.1. Ukázkový výpis ze směrovací tabulky koncového klienta je vidět v tabulce 3.

SSRC účastníka	**SSRC proudu dat**	**PayloadType**	**TargetSSRC Seznam cílových adresátů**	**SourceSSRC Původní zdroj**	**Confirmed**
68965	57889	FbpControlData	84535	68965	TRUE
68965	24478	MJPEG	84535	36528	TRUE
68965	78895	G.772	84535	36528	TRUE

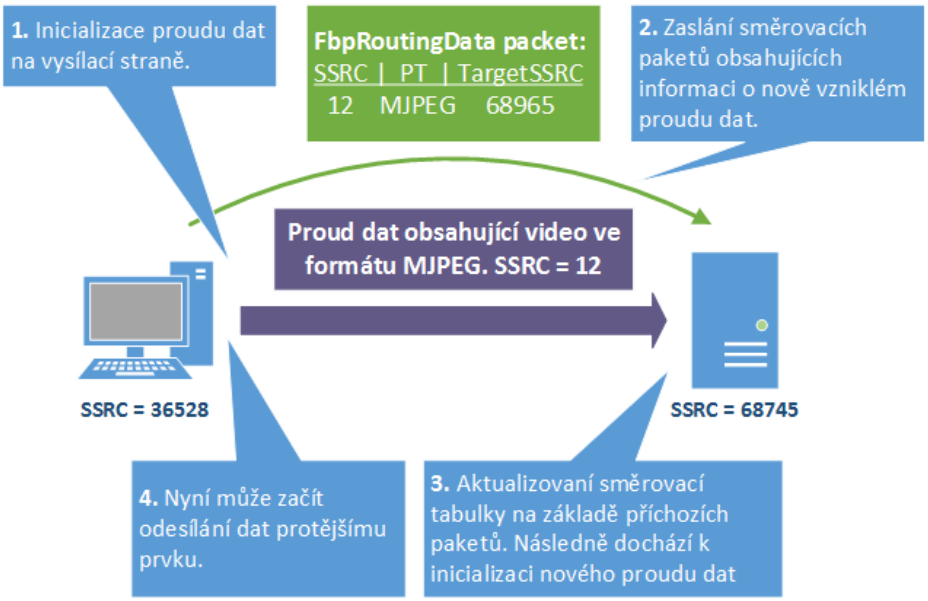
Tabulka 3: Ukázka směrovací tabulky

Jak je z tabulky vidět, tak poslední uzel v řetězci s identifikátorem 68965 zasílá našemu klientovi tři typy dat identifikované čísly 57889, 24478 a 78895. Posledním uzlem je nazvaný proto, že jako poslední prvek přeposílá data k nám. Jako koncoví klienti neznáme rozložení sítě a nevíme přesně, přes které prvky v síti k nám data přišla. Víme pouze, že poslední, kdo data přeposlal je uzel s identifikátorem 68965. Pro lepší pochopení je na obrázku 10 znázorněno rozložení všech prvků v této síti. V tabulce dále vidíme, že nám je zasíláno video ve formátu MJPEG a audio kódované pomoci kodeku G.772. Cílovým adresátem je náš klient s identifikátorem 84535 a multimediální data pochází od klienta s identifikátorem 36528. Všechny proudy dat jsou navíc potvrzeny a můžeme počítat s příjmem jednotlivých datagramů s užitečným obsahem.

Jak je zřejmé z předchozího odstavce, tak pro jakákoliv přenášená data existuje záznam ve směrovací tabulce. Pokud bude účastník vysílat například několik proudů videa, tak pro každý proud bude v tabulce příjemce uveden jeden záznam, který bude mimo jiné obsahovat informaci o cílových adresátech. Bez těchto směrovacích informací by příjemce dat nevěděl, jestli jsou data určena pro něho nebo pro jiného účastníka. Pokud by se na základě jedinečného identifikátoru obsaženého v datagramu nenašel ve směrovací tabulce žádný záznam, je příjemce tohoto datagramu povinný takový datagram zahodit.

6.4.2 Vznik směrovací tabulky

Směrovací tabulka příjemce vzniká na základě vysílaných proudů dat každého odesílatele. S každým nově založeným proudem dat se vygenerují potřebné směrovací informace a ty se v pravidelných intervalech zasílají jeho příjemci. Ten si na jejich základě aktualizuje svou směrovací tabulku, viz obrázek 9. Každému příjemci se tedy zasílají pouze ty informace, které jsou pro něj užitečné.



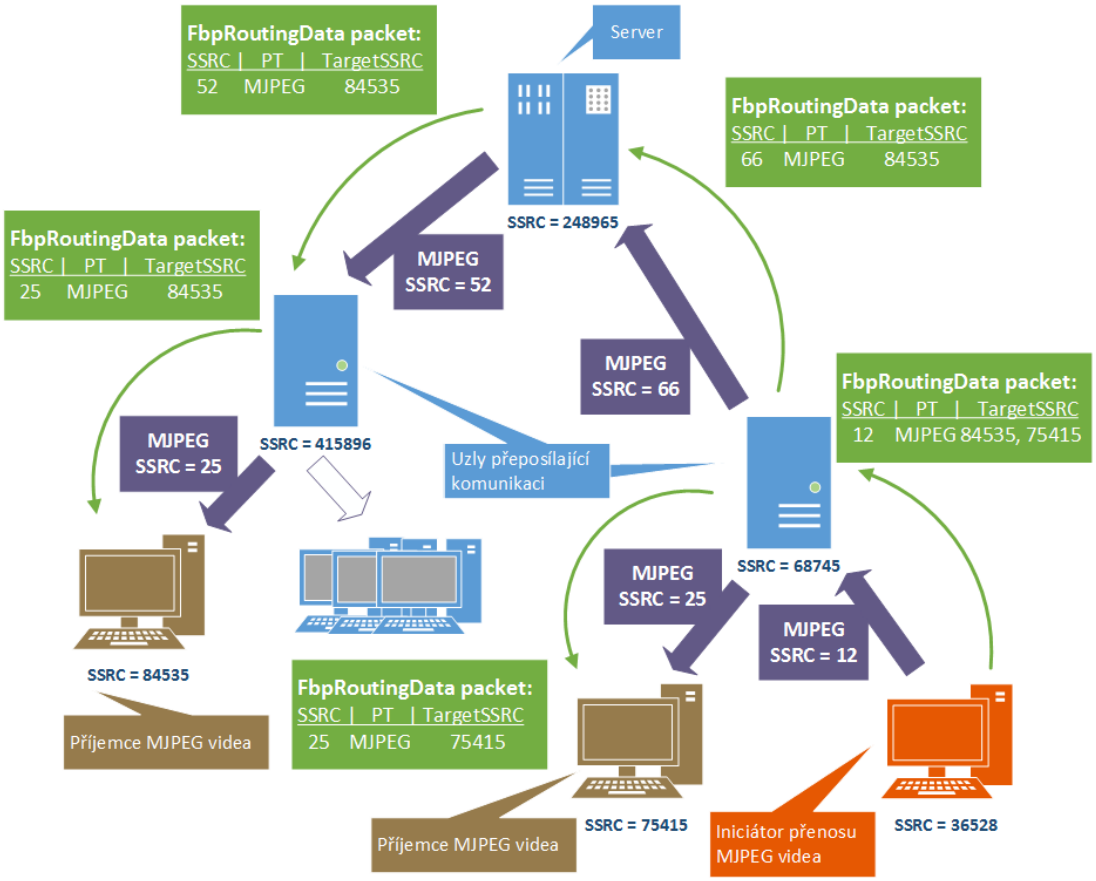
Obrázek 9: Proces aktualizace směrovací tabulky

Každému prvku v sítí se ovšem nezasílají stejné směrovací informace. Na základě typu spojení můžeme jednotlivé prvky rozdělit do dvou kategorii. První skupinu tvoří prvky, které se k nám připojily. Těm říkáme potomci. Pro ně vystupujeme jako server nebo jako uzel. Může jít například o klienty nebo o další uzly. Druhá skupina obsahuje vždy jen jeden prvek a to prvek, ke kterému jsme připojeni. Ten nazýváme rodičem. Může to být uzel nebo server. Server je navíc speciálním případem, který je kořenem stromu a nemá již žádného rodiče. Na základě těchto dvou kategorií jsou poté zasílány různé směrovací informace.

Směrem k potomkům se odesílá pouze směrovací tabulka obsahující naše proudy dat směřující k danému prvku. Takový příklad lze vidět v tabulce 3. Na základě těchto směrovacích informací následně prvek aktualizuje nebo inicializuje proud dat pro příjem datagramů. Směrem k rodiči odesíláme také informace o vysílaných proudech dat, které k němu směřují. Navíc se nadřízenému prvku zasílají informace o všech potomcích a jejich potomcích. Informace o potomcích obsahují pouze jedinečný identifikátor a způsob připojení. Nejsou zde obsaženy informace o vysílaných proudech dat. To by bylo zbytečné, protože pokud potomek zasílá data dále směrem k serveru, tak vzniká na uzlu nový proud dat, který je identifikován vlastním identifikátorem.

6.4.3 Směrování podle tabulky

Díky tomuto způsobu směrování zná každý nadřízený prvek všechny účastníky, kteří jsou v jeho podsíti. Kořen stromu následně ví o každém připojeném prvku ve ForceB síti. Nezná ovšem jeho přesnou polohu ale pouze následující uzel, tzv. next hop. Na základě těchto informací je možné obsah vždy doručit k cílovému adresátovi, aniž by tento účastník byl přímo připojený k serveru nebo konkrétnímu uzlu. Ukázku směrování v síti znázorňuje obrázek 10.



Obrázek 10: Ukázka směrování ve ForceB síti

Na obrázku je znázorněna síť, ve které vystupuje server, dva uzly a čtyři klienti. Klient s identifikátorem 36528 začne inicializovat přenos videa ve formátu MJPEG. Cílem jsou dva klienti s identifikátory 75415 a 84535. Klient vysílající video odešle aktuální směrovací tabulku směrem k uzlu, ke kterému je připojen. Ten ví, že klient s číslem 75415 je v jeho podsíti a inicializuje nový proud dat směrem k němu. Druhý z cílů již v podsíti není, a proto uzel inicializuje nový proud dat směrem k serveru. Server zná výchozí uzel ke každému klientovi a inicializuje proud dat směrem k uzlu s číslem 415896. Tento uzel již má informaci o cílovém prvku a tak inicializuje nový proud dat směrem ke klientu s

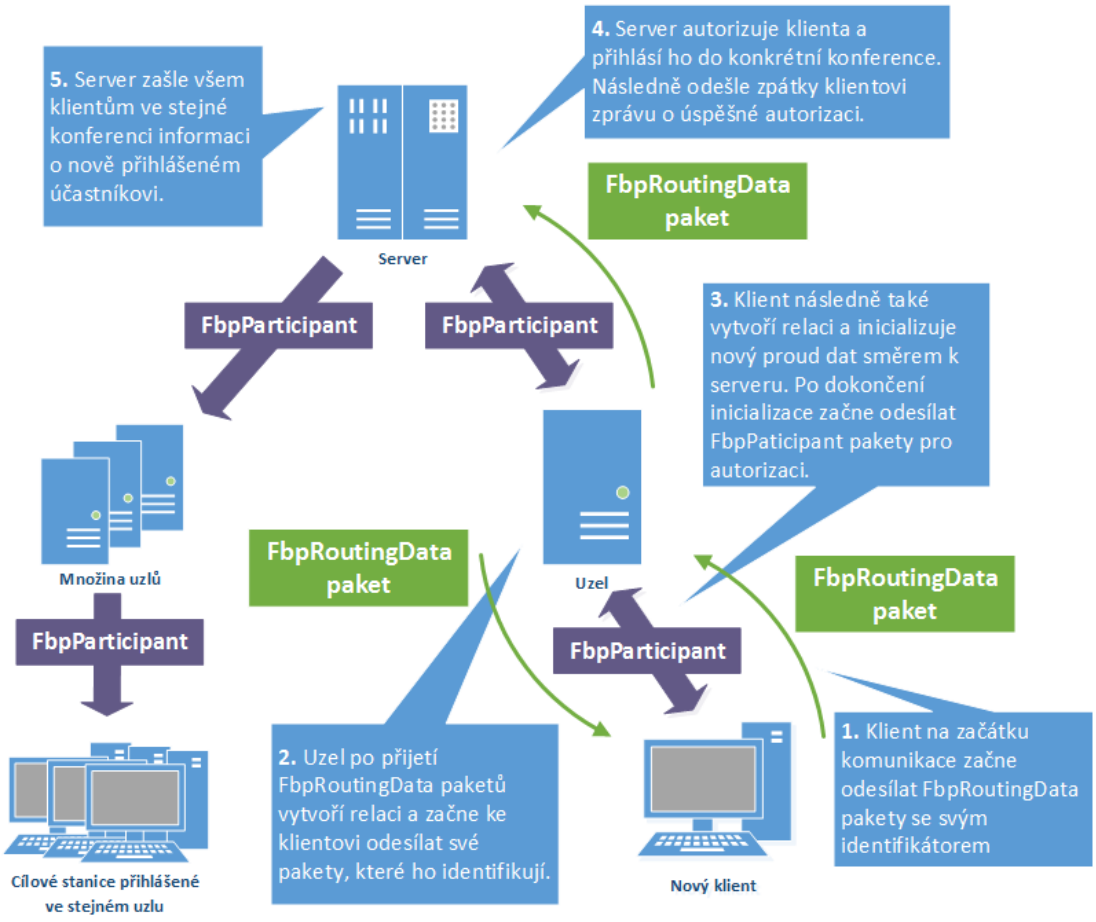
číslem 84535. Pokud by v jeho podsíti bylo více cílových klientů, inicializuje proud dat i k nim. Díky tomu data přes server fyzicky tečou pouze jednou a k přeposílání dochází až na samotných uzlech. To umožňuje lokální vytížení sítě rozložit na několik uzlů.

6.5 Přihlášení účastníků do skupiny

Pro přihlášení účastníka do sítě ForceB je nutné navázat komunikaci s některým z koncových bodů. V případě, že se do skupiny přihlašuje nový uzel, tak je koncovým bodem jiný uzel nebo server. V případě klienta, je koncovým bodem vždy některý z uzlů. Pro přenos datagramů je v tomto případě využit protokol UDP. Po navázání komunikace s koncovým bodem vzniká obousměrná relace, kterou inicializuje klient.

Pro navázání spojení začíná klient nebo uzel vysílat FbpRoutingData pakety v pravidelných intervalech. Ty obsahují jedinečný identifikátor, který si prvek vygeneruje při inicializaci. Jedná se o náhodné číslo v uint rozsahu. Prvek se tímto identifikátorem identifikuje v celé ForceB síti. Protější prvek po přijetí těchto paketů vytvoří obousměrnou relaci a začne v pravidelných intervalech také vysílat FbpRoutingData pakety s vlastním identifikátorem. Po přijetí tohoto datagramu inicializuje relaci i prvek inicializující spojení a komunikace na portu 23589 je navázána. V tento okamžik dochází k aktualizaci směrovacích tabulek směrem k serveru, který se díky tomu dozví o nově přihlášeném prvku do sítě ForceB. V tomto okamžiku ještě není takový prvek autorizován. V případě, že se do sítě připojil uzel, tak již žádná autorizace neprobíhá a slouží pouze pro přeposílání komunikace.

Pokud se do sítě připojil klient, tak po navázání relace s koncovým bodem začíná posílat FbpParticipant pakety. Ty slouží pro autorizaci klienta. Obsahují mimo jiné údaje CName, Pass a ConferenceId. Tyto pakety nejsou cíleny koncovému bodu, se kterým klient navázal komunikaci, ale přes něj putují až k samotnému serveru. Pro tento typ přenosu musí klient vytvořit nový proud dat a odeslat aktualizovanou směrovací tabulku koncovému bodu, ke kterému je připojen. Ten následně inicializuje proud dat směrem k serveru. Po dokončení inicializace proudu dat až k serveru, může klient začít odesílat autorizační pakety. Po přijetí paketů server provede autorizaci a vyšle klientovi zpět zprávu o úspěšné nebo neúspěšné autorizaci. Pokud je autorizace úspěšná, zařadí klienta na základě ConferenceId do konkrétního kurzu a všem přihlášeným klientům odešle zprávu o přihlášení nového účastníka. Celý proces přihlášení je zachycen na obrázku 11.



Obrázek 11: Proces autorizace klienta

7 Systém pro pořádání webinářů – ForceB

ForceB je systém určený k pořádání webových seminářů v prostředí internetu a univerzitní sítě. Umožňuje pořádat semináře až pro stovky účastníků, díky vlastnímu komunikačnímu protokolu ForceB. Cílem bylo navrhnout a implementovat nástroj pro vzdálenou výuku. Snahou bylo zvolit řešení postavené na nových technologiích, které bude snadným způsobem dále rozšiřitelné o podporu nových funkcí formou zásuvných modulů. Implementace systému navazuje na implementaci nástroje pro vzdálenou online výuku, který byl předmětem mé bakalářské práce.

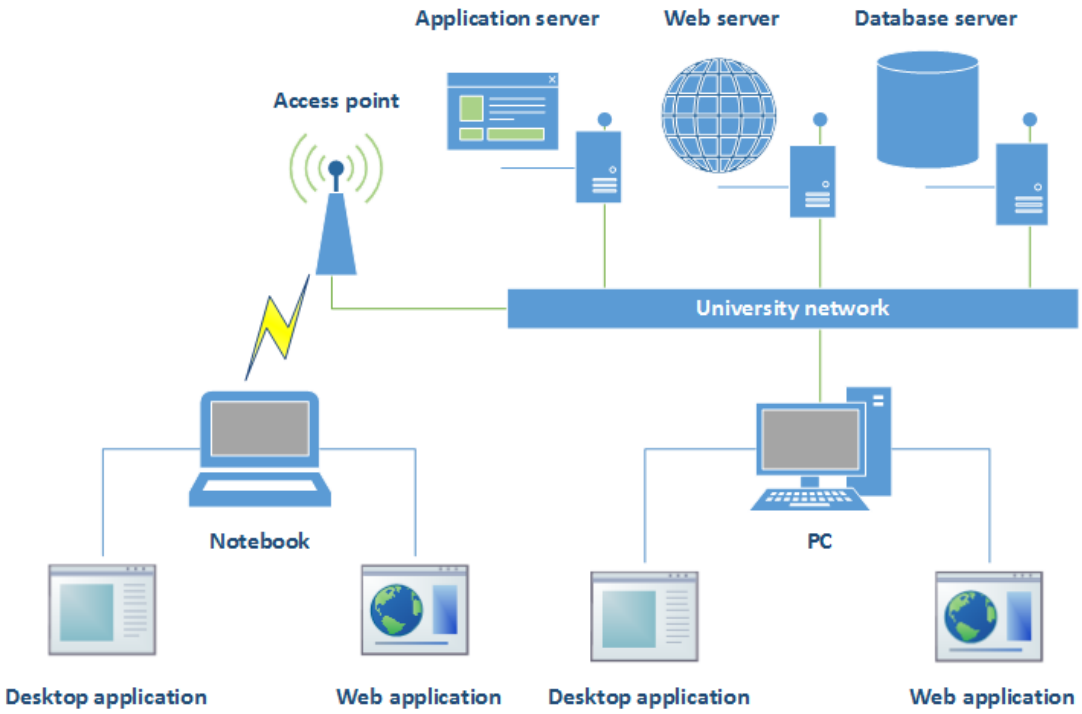
Mezi hlavní požadavky systému patří možnost pořádat online kurzy pro studenty. Kurzy jsou také snadno dostupné bez nutnosti instalace dodatečného softwaru nebo speciálního hardwaru. Jedná se o řešení SaaS, neboli software jako služba [13]. Systém nabízí možnost online výuky a spolupráce stejným způsobem, jako by účastníci seděli v jedné místnosti.

Jako klíčové vlastnosti systému ForceB byly zvoleny:

- Videokonference směrem k posluchačům.
- Telekonference mezi všemi účastníky.
- Chat.
- Elektronická tabule.
- Sdílení celé pracovní plochy monitoru.
- Sdílení okna konkrétní aplikace.

Systém není závislý na žádném softwaru třetích stran a obstarává veškeré funkce od serverové části až po samotné klienty. Skládá se z několika částí. Jsou zde tři servery. Databázový server uchovává data o jednotlivých studentech, kurzech a seriálech. Webový server slouží pro administraci jednotlivých kurzů, seriálů a studentů. Student se zde může zaregistrovat do daného kurzu a stáhnout si přiložené dokumenty. Posledním serverem je aplikační server. Jde o centrální prvek celého systému. Zprostředkovává komunikaci mezi jednotlivými uzly a zařazuje klienty do konkrétních kurzů.

Součásti celého systému jsou také dva klienti. Každý z klientů je určen pro rozdílnou platformu. Výchozím klientem je aplikace pro Windows desktop. Tato aplikace je určená především pro přednášející, kteří online kurzy pořádají. S její pomocí mohou posílat textové zprávy, zvuk a obraz z webové kamery ostatním účastníkům kurzu. Dále v závislosti na druhu výuky lze použít různé typy webových seminářů. Základní verze aplikace umožňuje sdílení obrazovky, sdílení okna konkrétní aplikace nebo sdílení prezentací vytvořených pomoci aplikace Microsoft PowerPoint [28]. Aplikace může rovněž sloužit jako tenký klient ke sledování kurzu. Dalším klientem je aplikace určená pro webové prohlížeče s podporou technologie Microsoft Silverlight [4]. Student díky této aplikaci nemusí instalovat žádné dodatečné aplikace do systému. Vše probíhá v okně webového prohlížeče. Schéma celého systému lze vidět na obrázku 12.



Obrázek 12: Schéma systému ForceB

7.1 Použité technologie

Systém ForceB byl vyvíjen v prostředí Microsoft Visual Studio 2012 [21] na platformě .NET. Serverová část běží jako služba systému Widnows nebo jako worker role ve Windows Azure [25, 26]. Pro implementaci desktopového klienta byla využita technologie WPF (Windows Presentation Foundation) [29]. Pro webového klienta byla zvolena platforma Microsoft Silverlight [20]. Webové rozhraní informačního systému je postavené na platformě ASP .NET [19] a o databázovou část se stará Microsoft SQL Server [36].

7.1.1 Windows Presentation Foundation

Windows Presentation Foundation (WPF) je grafický framework pro psaní Windows aplikací. Jedná se o podmnožinu .NET Frameworku od verze 3.0, který používá značkovací jazyk XAML. Technologie WPF je vestavěná do Windows Vista, Windows 7 a Windows Server 2008 a je dostupná pro Windows XP SP2 a Windows Server 2003. Díky XAMLu jsou od sebe odděleny funkčnost a vzhled aplikace. Cílem WPF je sjednotit poutavé uživatelské rozhraní, 2D a 3D grafiku, vektorovou a rastrovou grafiku, animace, vázání dat a audio a video. Předchůdcem WPF jsou Windows Forms (WinForms). WPF ale není technologie nahrazující WinForms, jedná se spíše o další, druhý způsob psaní Windows aplikací. Oproti WinForms ale nabízí nové možnosti hlavně v oblasti grafického zpracování aplikace, které ve WinForms nejsou možné.

7.1.2 Microsoft Silverlight

Silverlight je aplikační platforma vytvořená společností Microsoft, která je určena pro vývoj business a multimediálních aplikací (Rich Internet application - RIA). Tato platforma je určena pro tvorbu dynamického online obsahu a interaktivní práci s ním. Pomocí této platformy lze snadno kombinovat text, vektorovou i bitmapovou grafiku, animace a video. Podobně jako u technologie WPF se uživatelské rozhraní typicky definuje pomocí deklarativního programovacího jazyka XAML.

Plug-in je dostupný pro většinu webových prohlížečů (Internet Explorer, Firefox, Opera, Chrome, Safari) na platformách Widnows a Mac OS X. Je dostupný také pro Linux pod názvem Moonlight, vyvinutý společností Novell.

7.1.3 ASP .NET

ASP.NET je součást .NET Frameworku pro tvorbu webových aplikací a služeb. Je nástupcem technologie ASP (Active Server Pages). ASP.NET je založen na CLR (Common Language Runtime), který je sdílen všemi aplikacemi postavenými na .NET Frameworku. Koncept ASP.NET WebForms ulehčuje programátorům přechod od programování klasických aplikací pro Windows do prostředí webu.

7.1.4 Microsoft SQL Server

Microsoft SQL Server je relační databázový a analytický systém vyvinutý společností Microsoft. Dovoluje ukládat data ze strukturovaných, částečně strukturovaných a nestrukturovaných dokumentů, jako jsou obrázky a multimediální soubory, přímo v rámci databáze. SQL Server nabízí širokou škálu integrovaných služeb, které umožňují provádět další operace s daty, jako například dotazy, vyhledávání, synchronizaci, generování sestav a vytváření analýz. Umožňuje využívat data z vlastních aplikací vyvinutých na platformách Microsoft .NET a Visual Studio a z architektur orientovaných na služby (Service Oriented Architecture - SOA) a obchodních procesů prostřednictvím serveru Microsoft BizTalk Server.

8 Implementace systému ForceB

Tato kapitola je rozdělena do tří částí. První část tvoří implementace komunikačního protokolu systému ForceB. Jsou zde popsány nejdůležitější třídy, které implementují protokol FBP. Dále obsahuje některé procesy starající se o chod celého systému. Druhá část je zaměřena na implementaci samotného systému ForceB, kde jsou opět popsány nejdůležitější třídy a to jak na straně klienta, tak na straně serveru. Poslední část tvoří implementace informačního systému.

8.1 Implementace ForceB protokolu

ForceB protokol (FBP) je naimplementován sadou knihoven. Tato implementace se snaží o určitou abstrakci celého procesu zpracování dat. A to od jejich paketizace, odeslání a následné přijetí až po znovu sestavení. Před programátorem, který vytváří aplikaci nad touto implementací, skrývá veškerou problematiku směrování, přihlašování se do skupin a podobně. Při přenosu dat vytváří iluzi přímé komunikace se vzdáleným klientem.

Programátor pouze řekne, že chce odeslat data skupině cílových stanic a o nic víc se nestará. Práce jednotlivých vrstev, tedy třeba to, že se data rozdělí do několika paketů, jimiž se po cestě zabývá několik uzlů, nemusí programátora implementujícího aplikaci vůbec zajímat.

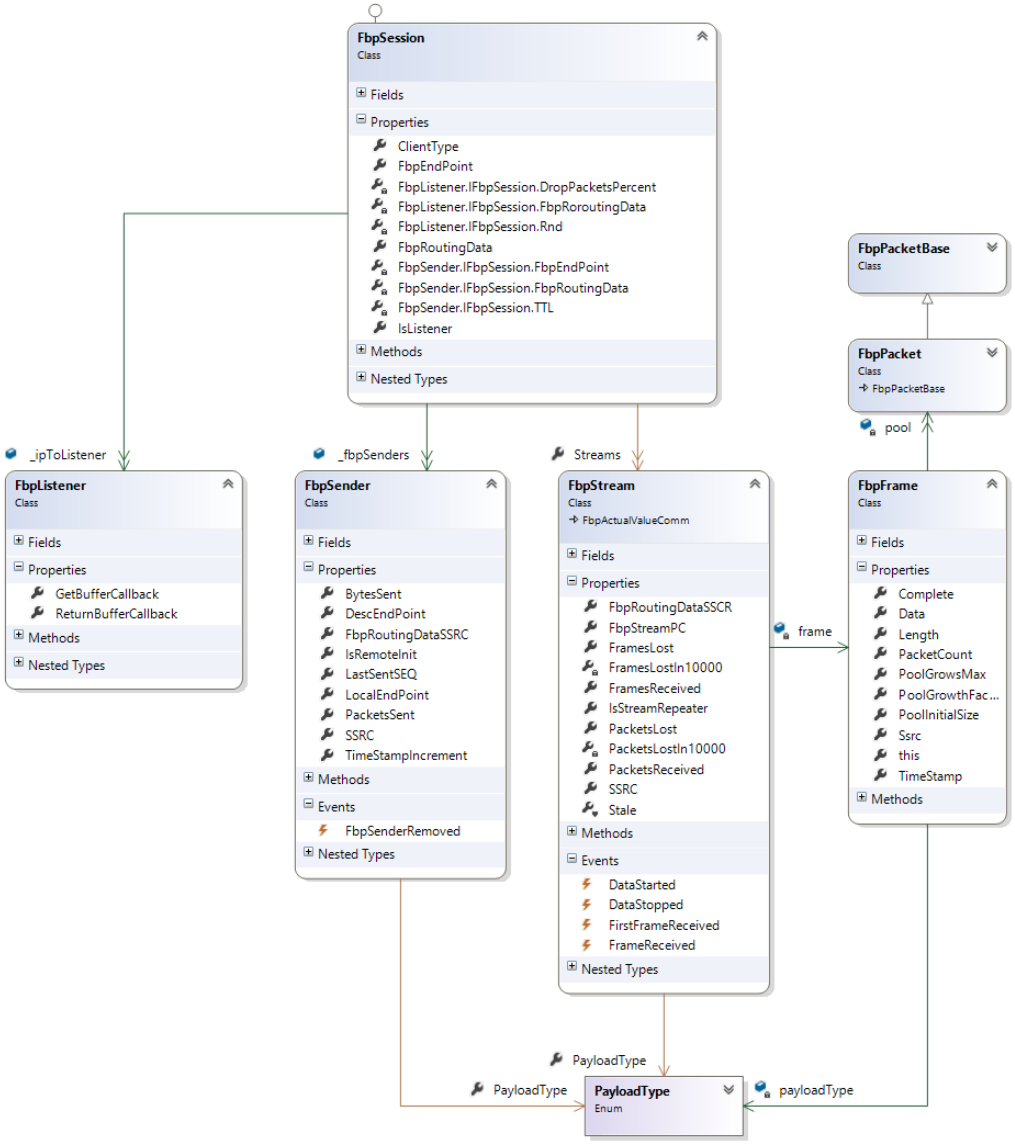
8.1.1 Architektura ForceB protokolu

Architektura ForceB protokolu je velmi rozsáhlá. V této kapitole budou popsány nejdůležitější třídy, které implementují tento protokol. Celý protokol je naimplementován v několika knihovnách, jejichž jmenný prostor začíná na ForceB.Desktop.Communication. Třídní diagram nedůležitějších tříd protokolu je znázorněn na obrázku 13.

Pro programátora využívajícího tuto knihovnu je nejdůležitější třída FbpSession, která zastřešuje většinu metod. Programátor bude tyto metody potřebovat pro navázání komunikace se sítí ForceB. Tato třída následně pracuje se třídami FbpListener pro naslouchání na koncovém bodě, FbpSender pro vytvoření nového proudu dat, FbpStream pro zpracování užitečného obsahu z tohoto proudu dat a třídou FbpFrame, která reprezentuje pole zaslaných bytů. Pro programátora je už prakticky skryta třída FbpPacket, jejichž instance tvoří samotný přenášený obsah rozdělený na malé díly o velikosti 1500B.

8.1.1.1 FbpSession: Jedná se o nejdůležitější třídu celého protokolu. Zastřešuje většinu logiky spojenou s inicializací spojení, přihlášením nových účastníků a směrováním. Tuto třídu využívá pro inicializaci spojení především klient. Server a jednotlivé uzly využívají třídu FbpSessionRepeater, která dědí z této třídy. FbpSessionRepeater implementuje navíc metody pro přeposílání datagramů jiným prvkům v sítí, viz kapitola 8.1.4.

Třída FbpSession implementuje několik rozhraní. Jedním z nich je FbpSender.IFbpSession, který obsahuje metody vztahující se k třídě FbpSender. Do každé vytvořené instance třídy FbpSender se v konstruktoru předává instance třídy FbpSession reprezentována



Obrázek 13: Diagram tříd ForceB protokolu

rozhraním FbpSender.IFbpSession. Druhým implementovaným rozhraním je FbpListener.IFbpSession, který obsahuje metody vztahující se k třídě FbpListener. Opět je instance třídy FbpSession předávána v konstruktoru této třídy. Každá z instancí obou tříd následně může volat metody implementované ve třídě FbpSession. Obě rozhraní jsou vidět v kódu 1 a 2.

```
public class FbpSender
{
  #region Interfaces

  internal interface IFbpSession
  {
    void AddBye(uint ssrc);
    uint NextSSRC();
    void Dispose(uint ssrc);

    FbpRoutingData FbpRoutingData { get; }
    System.Net.IPEndPoint FbpEndPoint { get; }
    ushort TTL { get; }
  }

  #endregion Interfaces
}
```

Výpis 1: Ukázka rozhraní FbpSender.IFbpSession

```
public class FbpListener
{
  #region Interfaces

  public interface IFbpSession
  {
    FbpStream GetStream(uint ssrc, IPEndPoint ip);
    void ProcessFbpControlData(FbpListener.ReturnBufferHandler listenerBufferCallback,
      uint ssrc, FbpDataBase fd, IPEndPoint descEndPoint);

    void RaiseInvalidPacketEvent(string msg);
    int DropPacketsPercent { get; }
    Random Rnd { get; }
    FbpRoutingData FbpRoroutingData { get; }
  }

  #endregion Interfaces
}
```

Výpis 2: Ukázka rozhraní FbpListener.IFbpSession

8.1.1.1.1 Proces inicializace a přihlášení: V případě, že tuto třídu inicializuje klient, tak do konstruktoru předá adresu cílového uzlu, ke kterému se chce připojit. Následně dojde k vygenerování jedinečného identifikátoru pro tohoto klienta. Tento identifikátor musí být jedinečný v celé síti ForceB. Pokud by se dále v procesu inicializace přišlo na to, že již existuje prvek se stejným identifikátorem, tak je klientovi zaslán oznamovací paket o duplicitním identifikátoru. Klient je poté povinen vygenerovat nový identifikátor. Číslo pro tento identifikátor je náhodně voleno z rozsahu uint. To znamená, že celkový rozsah je od 0 do 4294967295.

Po vygenerování identifikátoru dochází k vytvoření instance třídy FbpRoutingData. Ta reprezentuje směrovací informace daného klienta. Potom co je tato třída inicializována, spouští se nové vlákno, které v cyklu generuje a odesílá směrovací informace všem okolním prvkům. Třída FbpRoutingData je zděděná z třídy FbpDataBase. Tato třída obsahuje metody pro snadnou serializaci jednotlivých atributů do pole bytu k odeslání, viz kapitola 8.1.2.

Po odeslání těchto směrovacích informací se čeká na odpověď od nadřízeného prvku. Po přijetí odpovědi ve formě FbpRoutingData paketu je tento prvek zaregistrován. Následně mu jsou přiřazeny potřebné atributy a tímto je vytvořena obousměrná relace. Na konci tohoto procesu je vyvolána do aplikace událost FbpSessionConnected.

Aplikace následně volá na instanci třídy FbpSession metodu LoginParticipant, která vytvoří nový proud dat směrem k serveru a odešle FbpParticipantData pakety. Třída FbpParticipantData obsahuje informace sloužící pro autorizaci a přiřazení klienta do kurzu, viz kapitola 6.5. Po odeslání paketu se opět čeká na odpověď. Po přijetí odpovědi je celý proces inicializace dokončen a je vyvolána událost o úspěšném nebo neúspěšném přihlášení.

Pokud tuto třídu inicializuje uzel, tak po vytvoření obousměrné relace s nadřízeným prvkem nedochází k volání metody LoginParticipant. Místo toho dochází k vytvoření nové instance třídy FbpListener pro naslouchání na koncovém bodu pro připojení nových prvků do sítě. Může nastat ještě třetí varianta a to pokud třídu FbpSession inicializuje server. Ten při inicializaci neodesílá žádné směrovací FbpRoutingData pakety a pouze vytváří instanci třídy FbpListener pro naslouchání na koncovém bodě.

8.1.1.1.2 Proces zpracování směrovací tabulky: Směrovací tabulka je vytvářena na základě příchozích FbpRoutingData paketů, které jsou zasílány v pravidelných intervalech. Po přijetí takového paketu, od některého z prvků, dochází k jeho deserializaci a vytvoření instance třídy FbpRoutingData. Následně je ve dvou fázích tento paket zpracován. FbpRoutingData pakety obsahují informace o protějším prvku a také o jeho vysílaných proudech dat směrem k nám. V první řadě je volána metoda AddOrUpdateFbpRoutingData. Tato metoda na základě jedinečného identifikátoru SSRC zkontroluje, zda již takový prvek není v seznamu přihlášených účastníků. Pokud není, tak ho přidá a provede inicializaci. Pokud již přihlášený je, tak se zkontroluje, zda se shoduje IP adresa existujícího účastníka s IP adresou prvku, který zaslal tyto pakety. Jestliže by se neshodovaly, znamenalo by to, že se nám někdo snaží podstrčit podvrhnutou směrovací tabulku nebo že existuje přihlášený subjekt se stejným identifikátorem. V každém případě dojde k zaslání paketu, který oznámí duplicitní SSRC a k odpojení takového prvku.

V druhé fázi dochází k volání metody AddOrUpdateFbpStreamData. Práci této metody je aktualizování nebo přidání nového záznamu do naší směrovací tabulky na základě informací od protějšího účastníka. Pokud se při procházení směrovacích informací zjistí, že protější účastník vysílá nový proud dat, tak následuje volání metody AddOrUpdateStream. Tato metoda vytvoří pod daným SSRC identifikátorem novou instanci třídy FbpStream pro příjem dat. Následně dojde k vyvolání události FbpStreamAdded do apli-

kace. Tato událost obsahuje instanci třídy FbpStream, ze které bude aplikace získávat užitečný obsah zasílaný protějším prvkem.

Při příjmu datagramů s užitečným obsahem dochází k získání instance třídy FbpStream na základě SSRC identifikátoru a k předání takového datagramu ke zpracování a sestavení framu. Díky tomu, že si aplikace zaregistrovala metodu na událost FrameReceived, která je vyvolána po přijetí všech datagramu reprezentujících jeden frame, tak je vytvořena vazba mezi přijímanými daty a aplikaci.

8.1.1.2 FbpSender: Pokud chce aplikace odesílat jakákoliv data, musí mít instanci této třídy. Každá instance reprezentuje jiný proud dat. Pokud aplikace odesílá například zvuk a video, musí mít vytvořenou novou instanci pro oba zdroje dat. Novou instanci nevytváří přímo aplikace, ale volá metodu CreateFbpSender ze třídy FbpSession. Této metodě se předá seznam cílových účastníků, typ dat, který reprezentuje tento proud, a interní jméno zdroje dat.

Při vytváření nové instance se na základě cílových klientů dohledávají adresy koncových bodů, kterým mají být data odesílaná. Část kódu pro vyhledávání koncových bodů je vidět v kódu 3. Může dojít k situaci, kdy prvek není v přímém spojení s cílovými účastníky. K tomu dochází, pokud odesílá data koncový klient. Ten je vždy ve spojení pouze s uzlem a není tedy ve spojení s cílovými klienty, kterým chce data zasílat. V takovém případě, se vytvoří nový proud dat směrem k výchozímu uzlu, se kterým je účastník ve spojení a o ostatní přeposílání dat se nestará a nechá tuto problematiku následujícímu uzlu.

```
/// <summary>
/// Vytvori seznam participentu a ke kazdemu zvoli nextHop participenta pres ktereho se data
    dostanou k cili
/// </summary>
/// <param name="desFbpRoutingDataSSRC"></param>
/// <returns></returns>
private Dictionary<uint, List<uint>> GetNextHopForFbpRoutingDataList(List<uint>
    desFbpRoutingDataSSRC)
{
  Dictionary<uint, List<uint>> nextHopFbpRoutingDataAndDesFbpParticipent = new
      Dictionary<uint, List<uint>>();

  foreach (var fbpRoutingDataSSRC in desFbpRoutingDataSSRC)
  {
    if (fbpRoutingDataSSRC == _fbpRoutingDataMain.SSRC) // Nepridavam do seznamu pro
        prijemce sam sebe...!!
      continue;

    uint nextHopSSRC = GetNextHopForFbpRoutingData(fbpRoutingDataSSRC); // Pro
        konkretni cil dohledavam nextHop
    if (nextHopSSRC != 0)
    {
      if (!nextHopFbpRoutingDataAndDesFbpParticipent.ContainsKey(nextHopSSRC)) //
          Pokud jeste tekovy nextHop v seznamu neni, pridam jej
        nextHopFbpRoutingDataAndDesFbpParticipent.Add(nextHopSSRC, new List<uint>()
            { fbpRoutingDataSSRC });
```

```
            else
            { //Pokud jiz nextHop v seznamu je, jen k nemu pridam dalsiho ciloveho prijemce (Jde o
                next hop, pres ktery se k nemu dostanu)
              var desFbpParticipentList = nextHopFbpRoutingDataAndDesFbpParticipent[
                  nextHopSSRC];
              if (desFbpParticipentList == null)
                desFbpParticipentList = new List<uint>();
              desFbpParticipentList.Add(fbpRoutingDataSSRC);
            }
          }
        }

        // Pokud bude seznam prijemcu prazdy potom posilam data na next hop... (Data urcena pro
            server)
        if (desFbpRoutingDataSSRC.Count == 0 && _fbpRoutingDataNextHop != null)
          nextHopFbpRoutingDataAndDesFbpParticipent.Add(_fbpRoutingDataNextHop.SSRC, new
              List<uint>());

        return nextHopFbpRoutingDataAndDesFbpParticipent;
      }
```

Výpis 3: Vyhledávání koncových bodů

Jestliže vytváří nový proud dat uzel, tak se na základě cílových adresátů vytvoří nové proudy dat ke všem cílovým účastníkům, kteří jsou s tímto uzlem ve spojení. Pokud existují i účastníci, kteří nejsou v přímém spojení s tímto uzlem, tak se automaticky vytváří nový proud dat směrem k výchozímu uzlu, kterým může být i server, viz obrázek 10.

Každá nově vytvořená instance FbpSenderu znamená i nový záznam ve směrovací tabulce. Na jejím základě jsou potom generovány FbpRoutingData pakety, které informují okolní prvky o vytvoření nového proudu dat. Před tím, než je proces inicializace FbpSenderu dokončen, musí dojít k potvrzení každého proudu dat, viz obrázek 7.

Pro samotné odesílání dat stačí aplikaci zavolat metodu Send nad vytvořenou instancí třídy FbpSender. Této metodě se předává jako parametr buffer dat. Metoda následně vytvoří instanci třídy FbpFrame, která zaobaluje odesílaný buffer dat. Každému framu je přiřazen jedinečný identifikátor proudu dat, TimeStamp, pořadí odesílaného framu a mnoho dalších informací. FbpFrame se stará o rozdělení bufferu dat na jednotlivé pakety o velikosti maximálně 1500B. Každému paketu následně vytvoří potřebnou hlavičku a uloží jej do pole k odeslání. Po dokončení tohoto procesu se vybere socket pro přenos dat, kterému je předán frame s pakety k odeslání.

8.1.1.3 FbpListener: Třída se stará o příjem a zpracování nových datagramů. Celé zpracování je rozděleno do několika vláken tak, aby se využilo co možná nejvíce paralelizmu a nevzniklo při zpracování úzké hrdlo. Zpracování přijatých dat je rozděleno do tří okruhů. Každý z okruhů nejprve data získá z fronty, následně data zpracuje a pak je předá do následující fronty pro další zpracování.

První z okruhů se stará o přijetí datagramů ze sítě. Poté co data přijme, vloží je spolu s adresou koncového bodu do fronty pro další zpracování. Následně se celý cyklus opakuje, dokud není celá třída uvolněná. Výřez této metody je znázorněn v kódu 4.

```
private void ReceivePackets()
{
  BufferBase bc = null;

  while (true)
  {
    try
    {
      bc = GetBuffer();
      EndPoint ep;

      fbpNetworkListener.ReceiveFrom(bc, out ep);
      receivedPackets.Enqueue(new object[] { bc, ep });
      newPacket.Set();

      unchecked { pcPackets++; }
    }
    catch (ThreadAbortException) { }
    catch (PoolExhaustedException e)
    {
      LogEvent(e.ToString(), EventLogEntryType.Error, (int)FbpEL.ID.UnboundedGrowth);
      return;
    }
    catch (System.Net.Sockets.SocketException e)
    {
      ReturnBuffer(bc);

      if (e.ErrorCode != 10060)
      {
        Object[] args = new Object[] { this, new FbpEvents.
            HiddenSocketExceptionEventArgs((FbpSession)fbpSession, e) };
        EventThrower.QueueUserWorkItem(new FbpEvents.RaiseEvent(FbpEvents.
            RaiseHiddenSocketExceptionEvent), args);

        LogEvent(e.ToString(), EventLogEntryType.Error, (int)FbpEL.ID.Error);
      }
    }
    catch (Exception e)
    {
      ReturnBuffer(bc);
      LogEvent(e.ToString(), EventLogEntryType.Error, (int)FbpEL.ID.Error);
    }
  }
}
```

Výpis 4: Přijmutí nového datagramů a jeho vložení do fronty

Druhý okruh se v této implementaci protokolu stará o zpracování dat a předání dat do fronty pro další zpracování aplikace. Nejprve z fronty pro zpracování vytáhne datagram. Následně z jeho hlavičky zjistí, o jaký typ dat se jedná. Pokud se jedná o FbpControlData, volá metodu ProcessFbpControlData ze třídy FbpSession pro zpracování FbpControlDat. Může jít například o směrovací informace, statistická data a podobě. Podrobný seznam

je v kapitole 6.3.2. Pokud se na základě hlavičky datagramu zjistí, že jde o jakýkoliv jiný typ dat, tak se volá metoda GetStream ze třídy FbpSession. Tato metoda na základě SSRC identifikátoru z hlavičky datagramu vrátí instanci FbpStreamu pro zpracování daného proudu dat a předá jí nový paket pro zpracování, viz kapitola 8.1.1.4. Pokud by se instance třídy FbpStream pro tento identifikátor nenašla, dojde k zahození takového paketu. A to z důvodu možného útoku na protokol s cílem podvržení dat.

Třetím okruhem je fronta událostí vyvolávaných v aplikaci využívající tyto knihovny. Pokud je potřeba vyvolat událost pro aplikaci, tak se nezavolá přímo, ale zařadí se do fronty událostí. Do této fronty jsou přidávány všechny události, které mají komunikovat s aplikací. Tuto frontu následně obsluhuje další vlákno, které v aplikaci vyvolá potřebnou událost.

Díky těmto třem okruhům je zajištěno, že v žádném místě protokolu nevznikne úzké hrdlo, které by brzdilo zpracování nových dat. Největším rizikem je právě poslední část, kdy jsou data zpracovávaná aplikací. Špatnou implementací by se dalo lehce zpomalit celý proces přijímání a zpracování dat.

8.1.1.4 FbpStream: Instance této třídy je vytvořena pro každý příchozí proud dat. Nová instance třídy vzniká vždy na základě příchozí směrovací tabulky některého z účastníků. Na základě jedinečného identifikátoru SSRC v každém datagramu je volána konkrétní instance této třídy pomoci metody GetStream ze třídy FbpSesssion. Po vrácení instance třídy FbpStream se ji předají metodou PrcessPacket příchozí datagramy. Implementaci metody pro zpracování příchozích datagramů lze vidět v kódu 5. Třída se stará o sestavení framu z jednotlivých datagramů. Rozpoznává, zda nedošlo ke ztrátě nebo k přehození některých příchozích datagramů a následně na to reaguje. Po přijetí celého framu, který může reprezentovat například obrázek z webové kamery nebo část zvukové nahrávky, je vyvolaná událost FrameReceived.

```
internal virtual void ProcessPacket(FbpPacket packet)
{
  packetsLost += LostPackets(ref maxSeq, packet.Sequence);
  packetsReceived++;
  bytesReceived += packet.PayloadSize;
  lastReceivedSeq = packet.Sequence;

//  SendAck();

  uint packetTS = packet.TimeStamp;
  uint deltaTS = 0;

  if (packetsReceived != 1)
  {
    deltaTS = unchecked(packetTS − currentFrameTS);

    if (deltaTS > HALF_UINT_MAX) // pokud po odecteni bude vysoke cislo (pretekl) a jedna
        se o spozdeny packet
    {
      packetsReceivedLate++;
```

```
            returnBufferHandler(packet.ReleaseBuffer()); // Prisel packet ke staremu framu, uvolnim
                ho...
            return;
          }
        }

        if (frame == null)
        {
          frame = new FbpFrame(packet.PacketsInFrame, packetTS, returnBufferHandler);
          currentFrameTS = packetTS;
        }

        frame[packet.FrameIndex] = packet;

        if (frame.Complete)
        {
          OnFrameReceivedEvent(frame.Data);

          frame.Dispose();
          frame = null;

          currentFrameTS++;
        }
      }
```

Výpis 5: Proces zpracování přijatého datagramu

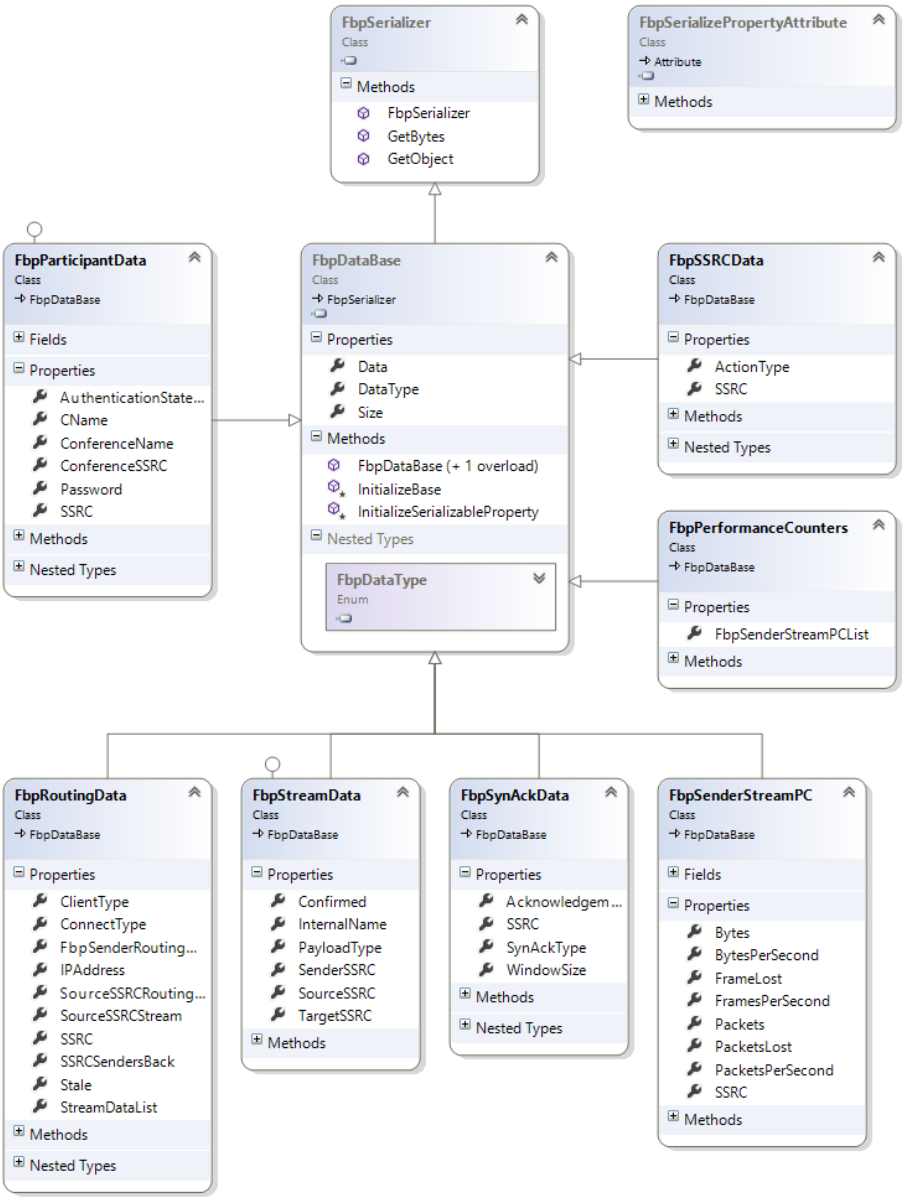
Pro každou instanci třídy jsou vytvářena statistická data, která můžou složit pro monitorování komunikační linky. Na základě těchto údajů je následně upravována komunikace mezi oběma klienty.

8.1.2 Implementace FBP Control Data paketů

V této kapitole bude popsána implementace FBPCD paketů. Budou zde popsány třídy, které reprezentují jednotlivé typy paketů a base třídy, starající se o serializaci a deserializaci dat jednotlivých property. Každý z paketů obsahující režijní data je reprezentován vlastní třídou pojmenovanou typem paketu. Například pro směrovací pakety je třída pojmenována FbpRoutingData, pro potvrzovací pakety je třída pojmenována FbpSynAckData a podobně. Diagram tříd zobrazující vazby mezi jednotlivými třídami je na obrázku 14.

Třída FbpDataBase je rodičem pro všechny třídy, které reprezentují jednotlivé typy FBPCD paketů. Tato třída se spolu s třídou FbpSerializer stará o binární serializaci a deserializaci property. Serializovány jsou pouze ty property třídy, které mají nastaven atribut na FbpSerializeProperty. K serializaci dochází při zavolání property Data, která vrací pole bytu serializovaného objektu.

Při serializaci se využívá reflexe platformy .NET. Třída FbpDataBase si nejprve načte pole všech property s atributem FbpSerializeProperty. Poté je v cyklu prochází a získá datovou hodnotu každé property. Následně vola metodu GetBytes z rodičovské třídy FbpSerializer, která na základě datového typu převede datovou hodnotu property na



Obrázek 14: Diagram tříd zobrazující vazby mezi FBPCD třídami

pole bytu. Po získání pole bytu je vytvořena struktura, do které se zapíše datový typ, velikost pole bytu a samotné pole bytu reprezentující datovou hodnotu, viz kapitola 6.3.1. Potom co jsou všechny property převedeny do potřebné struktury, jsou vloženy za sebe a první byte nové struktury říká o jaký typ FBPCD paketu jde.

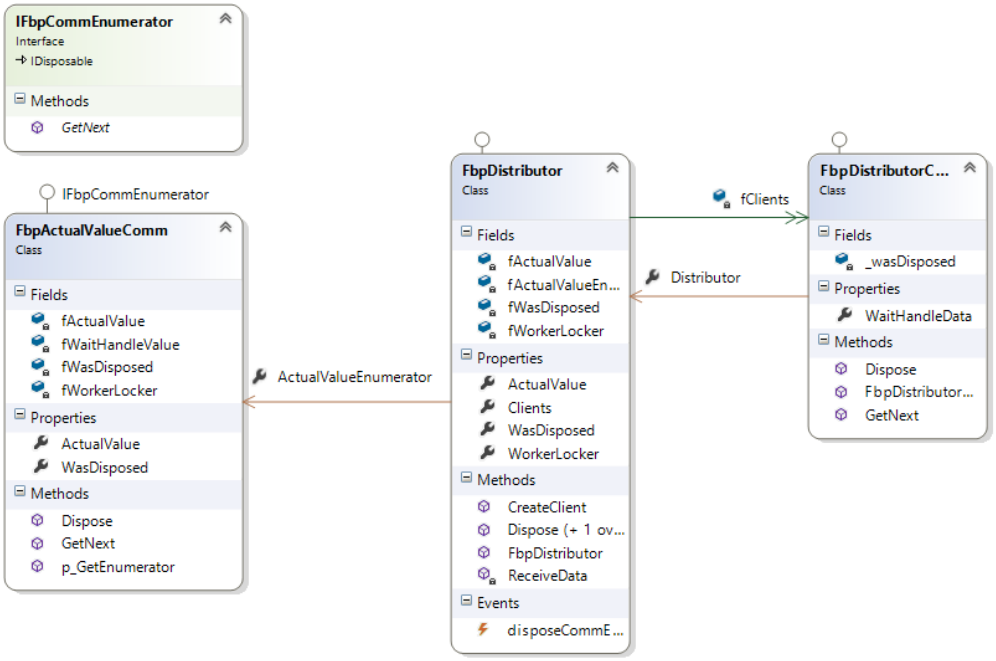
Při deserializaci pole bytu na potřebný datový typ se vychází z předem domluvené datové struktury. Při předání pole bytu do konstruktoru třídy FbpDataBase se provede částečná deserializace. Z vytvořené instance této třídy je následně možné zjistit typ FBPCD

paketu a při následném vytváření instance potřebné třídy (reprezentující FBPCD paket) se do konstruktoru předá buď pole bytu nebo instance třídy FbpDataBase, ve které již došlo k částečné deserializaci. Následně dojde k deserializaci zbylých property a nastavení jejich hodnoty v instanci této třídy.

Díky vlastnímu způsobu serializace jednotlivých tříd se ušetří oproti binární serializaci platformy .NET více jak polovina místa. Tato výhoda je patrná především při přenášení režijních dat FBP protokolu, kde se díky ušetřené velikosti každého zaslaného paketu ušetří i přenosová šířka pásma.

8.1.3 Implementace knihovny CommEnumerator

Třídy z knihovny se jmenným prostorem ForceB.Desktop.Communication.CommEnumerator složí pro obecnou práci s daty a jejich distribuci. Obstarávají ukládání vygenerovaných dat a jejich následnou distribuci okolním částem systému. Odebírat data z jednoho zdroje může pomoci těchto tříd libovolné množství zaregistrovaných příjemců. Každý z příjemců odebírá jenom takové množství dat, která stačí zpracovat. Mezi příjemci můžou být odběratelé, kteří zpracují všechna distribuovaná data ale také příjemci, kteří zpracují jenom polovinu nabízených dat. Jednotliví příjemci se navzájem nijak neomezují a nezpomalují. Třídní diagram tříd starající se o distribuci dát je znázorněn na obrázku 15.



Obrázek 15: Diagram tříd zobrazující třídy v knihovně CommEnumerator

Při použití této knihovny třída generující data dědí ze třídy FbpActualValueComm. Generátorem dat může být libovolná třída, která chce data předávat více odběratelům. V aplikaci ForceB se například tato knihovna využívá pro distribuci obrazu z webové

kamery, obrazu z prezentace nebo zvuku. Využívá se na straně generující data, i na straně klientů, kde jsou data přijímána ze sítě ForceB a dále distribuována pro zpracování a zobrazení uživateli. V implementaci ForceB protokolu je například tato knihovna použita ve třídě FbpStream, která dědí ze třídy FbpActualValueComm. Díky tomu třída FbpStream slouží jako obecný generátor dat a lze ji předat do třídy FbpDistributor pro další distribuci.

Instance třídy FbpActualValueComm je předávána v konstruktoru třídě FbpDistributor, kde se s inicializaci spouští nové vlákno. Toto vlákno v cyklu odebírá aktuální hodnotu z instance třídy FbpActualValueComm pomoci metody GetNext. Třída FbpDistributor dále obsahuje metodu CreateClient, která po zavolání vytvoří a vrátí instanci třídy FbpDistributorClient. Tuto metodu pro získání instance třídy FbpDistributorClient volá každý odběratel, který chce s daty pracovat. Aby byl celý koncept funkční, tak každý odběratel musí mít vytvořenou novou smyčku ve vlastním vlákně, ze které bude odebírat a dále zpracovávat data z instance třídy FbpDistributor.

8.1.4 Implementace knihovny FbpRepeater

Tato knihovna obsahuje pouze jednu třídu FbpSessionRepeater. Tato třída slouží pro přeposílání datagramů mezi jednotlivými uzly nebo mezi uzlem a klienty. Třída rozšiřuje funkce třídy FbpSession, ze které je zděděná, o metody sloužící k přeposílání datagramů. Instanci této třídy vytváří zejména server a jednotlivé uzly sítě ForceB. Tato knihovna obstarává veškerou práci spojenou s problematikou přeposílání datagramů.

Díky tomu, že třída FbpStream dědí ze třídy FbpActualValueComm, může být použita jako obecný generátor dat, viz kapitola 8.1.3. Pokud se na základě směrovacích dat při inicializaci nového proudu dat v implementaci FBP protokolu zjistí, že daný prvek není cílovým adresátem, tak se vyvolá událost FbpStreamAddedRepeater, kterou zpracovává právě instance třídy FbpSessionRepeater. Metoda obsluhující tuto událost vytvoří na základě směrovací tabulky nové proudy dat k cílovým adresátům a pro každého adresáta vytvoří z FbpDistributoru novou instanci třídy FbpDistributorClient. Instance třídy FbpStream, do které budou přicházet nové datagramy, bude automaticky distribuovat tyto datagramy do potřebných instancí tříd FbpDistributorClient. Odtud již budou odebírány a odesílány do vytvořených proudu dat k cílovým adresátům. Díky tomuto algoritmu mohou být data distribuována libovolnému množství klientů s různou šířkou přenosového pásma. Každý z klientů si potom odebírá jenom takový počet snímky (například z web kamery nebo prezentace), kolik jich je schopen přijímat.

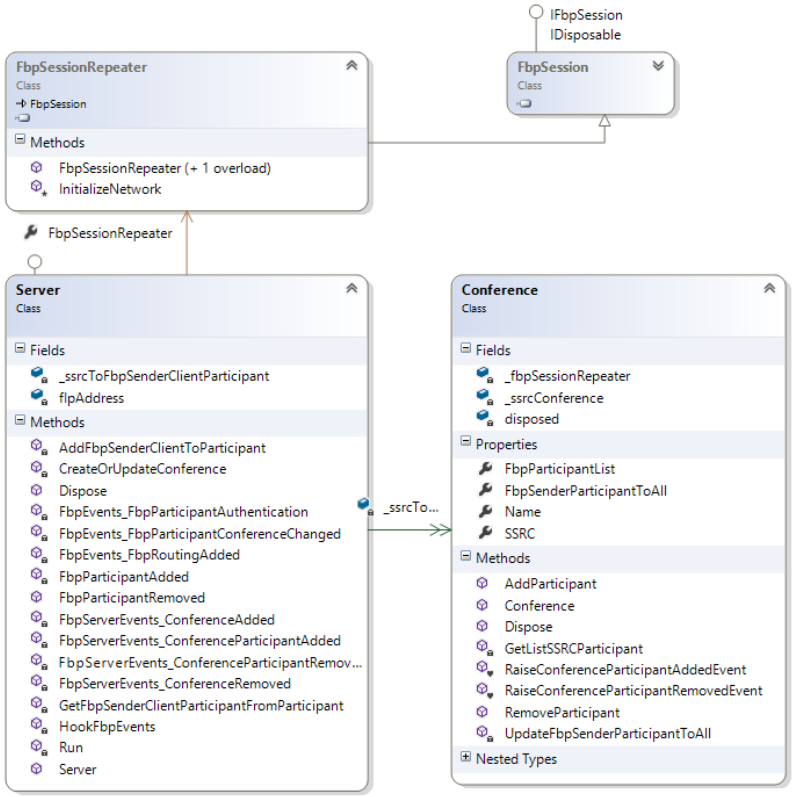
8.2 Implementace aplikační části systému ForceB

V této kapitole bude popsána implementace systému pro pořádání webových seminářů ForceB. Systém se skládá z několika částí. Je zde aplikační server, který se stará o přeposílání dat mezi uzly, o autorizaci klientů a o jejich zařazení do jednotlivých kurzů. Dále jsou zde uzly, které jsou koncovými body pro připojení klientů a starají se o přeposílaní komunikace mezi jednotlivými klienty a serverem. Do systému dále přistupují samotní klienti. Jde o dvě aplikace určené pro koncové uživatele. První z nich je desktopový klient určený pro pořádání webových seminářů. Druhým je klient spustitelný v internetovém

prohlížeči vytvořený na platformě Microsoft Silverlight. Poslední část tvoří informační systém, který zastřešuje celý projekt a slouží pro administraci kurzů a studentů.

8.2.1 Aplikační server

Aplikační server je mozkem celé sítě ForceB. Jedná se o nejdůležitější část síťové komunikace. Stará se o přeposílání dat mezi jednotlivými uzly, provádí autorizaci, zařazení klientů do jednotlivých kurzů a mnoho dalšího. Server může běžet jako služba systému Windows nebo jako worker role ve Windows Azure. Jádro serveru je v knihovně se jmenným prostorem ForceB.Desktop.Server. Třídní diagram serveru je na obrázku 16.



Obrázek 16: Diagram tříd zobrazující aplikační server

Serverová část pro založení ForceB sítě vytváří instanci třídy FbpSessionRepeater. V konstruktoru této třídy je předán koncový bod, na kterém bude server naslouchat příchozí spojení. Po inicializaci sítě si server zaregistruje potřebné události pro administraci nových klientů a kurzů. Poté co je vyvolána událost o přihlášení nového klienta, provede server autorizaci a následně klienta zařadí do některého kurzu. Pokud klient nemá definován žádný kurz, je vytvořen nový kurz na základě jeho jedinečného identifikátoru. Správu kurzu má na starosti třída Conference.

O přeposílaní dat mezi uzly se nestará přímo server ale instance třídy FbpSessionRepeater, viz kapitola 8.1.4. Tato třída řeší veškeré věci spojené s přeposíláním dat. Pokud je směrem k serveru vytvořen nový proud dat a server není uveden jako cílový prvek, tak se na základě cílových adresátů dohledají ve směrovací tabulce výchozí uzly pro každý z cílů a ke konkrétním uzlům se vytvoří nové proudy dat. Následně dochází ke stejnému procesu směrování na jednotlivých uzlech. Zde také dochází k dohledání cílových stanic a k vytvoření nových proudu dat k těmto cílovým stanicím. Datagramy směřující k serveru jsou poté vkládány do proudů dat směrujícím k potřebným uzlům a odtud následně dochází ke vkládání těchto datagramů do proudů dat inicializovaným k cílovým stanicím, viz obrázek 10.

8.2.2 Uzel

Uzly slouží ve ForceB síti k přeposílání komunikace mezi klienty, případně mezi klienty a serverem. Jedná se o klíčové prvky v celé komunikaci. Dokáží rozložit zátěž, kterou by jinak musel zpracovat server. S každou novou instancí uzlu je možné připojit desítky nových klientů. Na základě reálných testů byl jeden uzel schopen obsloužit až 40 klientů. Každému z klientů při tom uzel zasílal datový tok 2Mbit za sekundu.

Jednotlivé instance uzlů mohou být spuštěny podobně jako instance serveru. Může jít o službu systému Windows nebo o worker roli v Windows Azure. Díky třídě FbpSessionRepeater nemusí uzel implementovat žádnou další funkčnost. Po vytvoření instance této třídy je uzel zapojen do ForceB sítě a naslouchá na dané IP adrese a portu. V tuto chvíli se k němu mohou připojit klienti. Třídy implementující ForceB protokol se postarají o všechno ostatní. Vyřeší aktualizaci směrovacích tabulek, přeposílání dat, přihlášení klienta do skupiny a další.

Pokud je ForceB sít provozována v rámci platformy Windows Azure, tak je přidání nových instancí uzlů velmi snadné. Vše se konfiguruje ve webovém prohlížeči na portálu Widnows Azure. Pokud dojde k přetížení sítě, lze snadno přidat novou instanci workder role. Podrobněji se této problematice věnuje kapitola 9.2.2.

8.2.3 Desktop klient

Desktopový klient je aplikace určená pro systém Windows Vista a novější. Slouží především pro uživatele, kteří chtějí webové semináře pořádat. Klient má k dispozici funkce pro odesílání textových zpráv, odesílání obrazu z webové kamery, odesílání zvuku a další. Pro pořádání webových seminářů jsou v aplikaci k dispozici zásuvné moduly. Každý zásuvný modul je určen pro jiný typ webového semináře. K dispozici jsou tři zásuvné moduly. První slouží ke sdílení pracovní plochy monitoru, druhý ke sdílení okna konkrétní aplikace a se třetím je možné sdílet prezentace vytvořené v aplikaci Microsoft PowerPoint.

Pod pojmem zásuvný modul si můžeme představit dll knihovnu, která implementuje potřebná rozhraní a je uložená v adresáři pro zásuvné moduly. Jednotlivé zásuvné moduly je možné vytvářet bez zásahu do zdrojového kódu samotné aplikace. Aplikaci ForceB je možné díky těmto modulům dále rozšiřovat o nové funkce aniž by programátor musel

znát architekturu protokolu nebo principy zpracování dat v samotné aplikaci. Postup pro implementaci nového modulu je vysvětlen v kapitole 8.2.3.4.

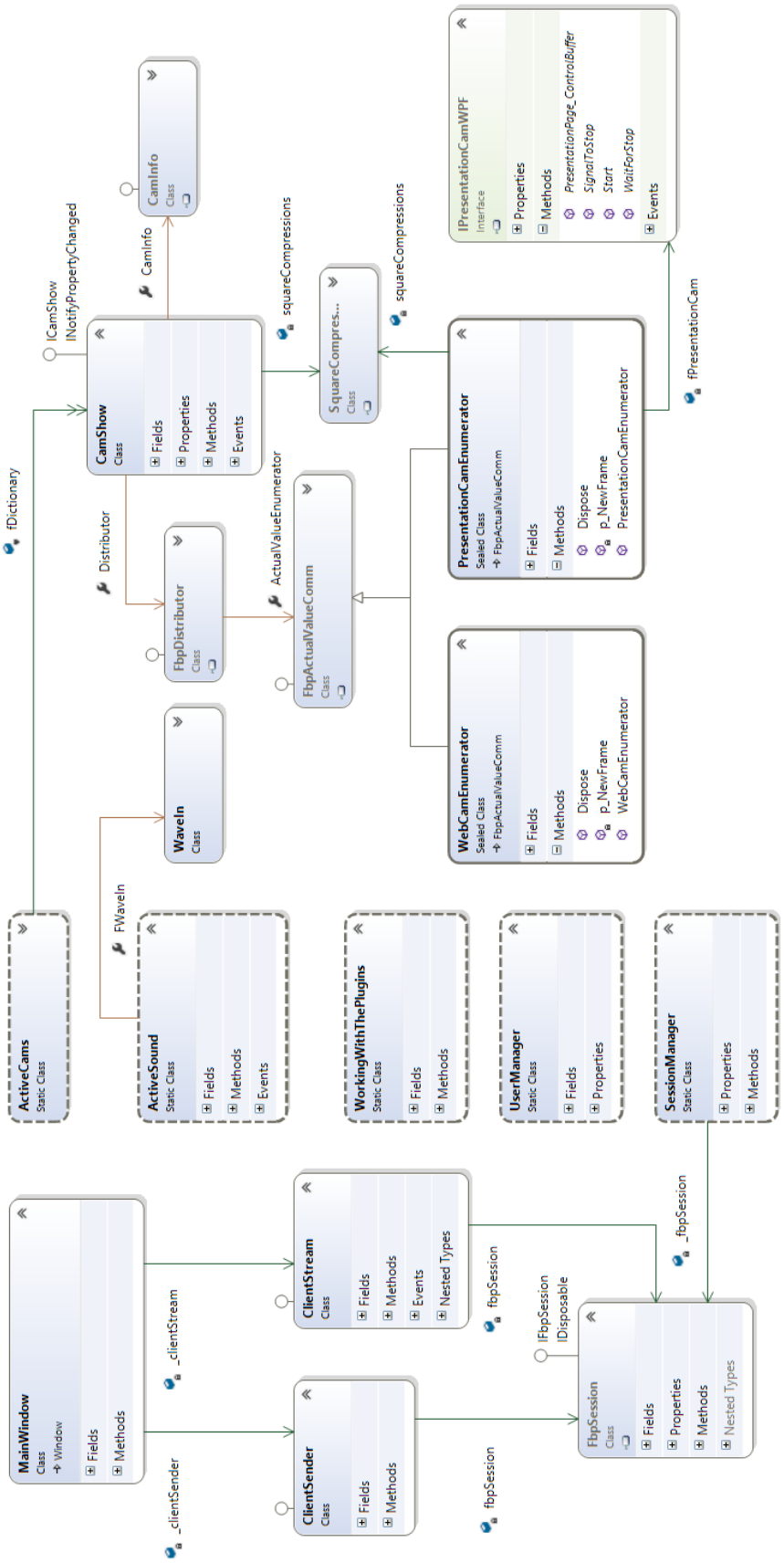
8.2.3.1 Architektura desktopového klienta Desktopový klient je vyvíjený na platformě .NET s využitím technologie WPF. Díky této technologii je aplikace oproti původní verzi napsané pod Windows Forms daleko svižnější. Původní problém s pomalým vykreslováním videa na celou obrazovku ve Windows Froms, již žádným problémem není. Zjednodušená architektura desktopového klienta je znázorněná pomocí třídního diagramu na obrázku 17.

Implementaci aplikace můžeme rozdělit do tří částí. První část tvoří třídy obstarávající generování videa, zvuku a obrazu pro samotný webový seminář. O tuto část se starají třídy ActiveCams, ActiveSound, CamShow, WebCamEnumerator, PresentationCamEnumerator a mnoho dalších. Druhou část tvoří třídy starající se o komunikaci se sítí ForceB. Zde spadají třídy jako je FbpSession, SessionManager, ClientSender a ClientStream. Do třetí částí můžeme zařadit třídy pro načítání zásuvných modulů, třídy starající se o kompresi audia a videa nebo třídy pro práci s nastavením. Nejdůležitější procesy jsou popsány v následujících kapitolách.

8.2.3.2 Inicializace klienta a připojení do ForceB sítě Po spuštění desktopového klienta dochází k inicializaci potřebných tříd a k načtení všech dostupných zásuvných modulů. Pro práci se zásuvnými moduly slouží třída Plugins. O načtení metadat k jednotlivým modulům se používá metoda GetAttributeFromPluginPresentation.

Po dokončení inicializace dochází k zobrazení dialogového okna pro přihlášení. Po zadání přihlašovacích údajů od klienta je inicializována statická třída SessionManager, která vytvoří instanci třídy FbpSession. V tuto chvíli knihovny implementující FBP protokol naváží komunikaci s ForceB sítí a provedou autorizaci klienta na základě zadaných přihlašovacích údajů. Pokud je klient zaregistrován do některého kurzu, dojde k jeho přihlášení do tohoto kurzu. Po úspěšné autorizaci dojde k vyvolání události FbpParticipantAdded. Tato událost obsahuje informace o přihlášených klientech ze stejného kurzu. Seznam přihlášených klientů se zobrazí v aplikaci.

8.2.3.3 Spuštění webového semináře Spuštění webového semináře může být inicializováno na straně prezentátora anebo na straně jednotlivých účastníků, kteří obraz z tohoto webového semináře budou přijímat. V obou případech dochází k volání podobných tříd. Díky implementaci jednotného rozhraní pro všechny zásuvné moduly se s instancemi těchto modulů pracuje jednotně v rámci celé aplikace. Při spuštění webového semináře se instance zásuvného modulu vytváří jak na straně prezentátora, tak na straně příjemce. A to z důvodů zpětné vazby, kdy i příjemce může zasahovat do webového semináře prezentátora. Například z důvodu kreslení na obrazovku prezentátora nebo přeskakování jeho slajdů. Obě dvě strany musejí ale pracovat se stejným zásuvným modulem. Pokud by se stalo, že klient nemá potřebný zásuvný modul nainstalován, tak se díky jednotné práci s rozhraním může obraz z prezentace takovému klientovi alespoň zobrazovat. Nemá ale možnost zasahovat do webového semináře.



Obrázek 17: Diagram tříd zobrazující nejdůležitější třídy desktopového klienta

Pokud webový seminář inicializuje prezentátor, tak nejprve dochází k inicializaci potřebného zásuvného modulu pro zvolený typ webového semináře. Pro vytvoření instance této třídy je volána metoda CreateInstanceFromPluginPresentation ze statické třídy Plugins. Následně dochází k volání metody StartNewPresentationCam ze statické třídy ActiveCams, které se předá instance zásuvného modulu reprezentována rozhraním IPresentationCamWPF.

Třída ActiveCams udržuje ve slovníku všechny instance zásuvných modulů a webových kamer. Slouží pro jejich inicializaci a také uvolnění. Metody z této třídy jsou volány ne jenom pro inicializaci na straně prezentátora ale i na straně příjemce. Pro ně jsou ale volány odlišné metody.

Následně je vytvořena instance třídy PresentationCamEnumerator, které je v konstruktoru předána instance zásuvného modulu. Tato třída dědí ze třídy FbpActualValueComm a obaluje obecnou práci se zásuvnými moduly. Každý nový snímek z webového semináře, který zásuvný modul vygeneruje, je v poli bytu předán rodičovské property ActualValue. Právě díky třídě FbpActualValueComm se ze třídy PresentationCamEnumerator stává generátor dat, který je dále předán do instance třídy FbpDistributor. Tato třída se stará o distribuci dat instancím třídy FbpDistributorClient. Jedná z instancí třídy FbpDistributorClient je vytvořená pro odesílání dat do ForceB sítě. Druhá může být vytvořena pro zobrazování obrazu u prezentátora. Například pro zobrazení vlastní webové kamery a podobně.

Každá instance FbpDistributorClienta si z instance třídy FbpDistributor bere jenom takové množství dat, které stačí zpracovat. Třídy pro distribuci dat se využívají i na jiných místech v aplikaci a v třídách implementujících ForceB protokol.

8.2.3.4 Zásuvné moduly a jejich implementace

Zásuvné moduly souží pro pořádání různých typu webových seminářů. S každým zásuvným modulem dostává aplikace nové funkce a možnosti výuky. Samotná aplikace zprostředkovává základní funkce systému jako je inicializace, přihlášení do sítě ForceB, přenos obrazu z webkamery nebo prezentace a další. Obraz pro prezentaci je ale generován v zásuvném modulu a až poté je předán aplikaci pro odeslání všem účastníkům. Zásuvný modul také může implementovat různé nástroje pro ovládání webového semináře a to na straně prezentátora i na straně účastníka.

V zásuvném modulu pro sdílení plochy monitoru jsou například k dispozici nástroje pro ovládání počítače prezentátora z pozice účastníka. Další funkci je kreslení a možnost označení místa na obrazovce prezentátora pro lepší vysvětlení. Zásuvný modul pro sdílení prezentací má například nástroje pro přechod mezi jednotlivými snímky prezentace. Jiný zásuvný modul může mít funkce zaměřené pro práci se vzorci a podobně. Vše záleží pouze na vývojáři, který tyto zásuvné moduly programuje.

Zásuvné moduly pro aplikaci ForceB může vyvíjet každý programátor využívající platformu .NET. Moduly jsou do aplikace přidávány formou dll knihoven. Každý modul musí implementovat potřebná rozhraní. Rozhraní slouží pro zobecnění práce s každým zásuvným modulem. Obsahuje metody pro spuštění zásuvného modulu a metody pro signalizování ukončení běhu modulu. Rozhraní dále obsahuje událost NewFrame, kterou vyvolává zásuvný modul, pokud chce odeslat nový snímek z prezentace všem účastníkům

v kurzu. Dále musí modul implementovat metodu PresentationPageControlBuffer, která je vyvolávána aplikací po přijetí dat od některého z účastníků. Pomocí této metody následně dochází ke zpětné vazbě od účastníků, kteří mohou do kurzu zasahovat. Na vývojáři modulu již je, jaké funkce pro ovládání webového semináře uvolní. Rozhraní potřebné pro implementaci lze vidět v kódu 6.

```
public interface IPresentationCamWPF
{
  uint SSRC { get; set; }

  PresentationBasePage fPresenationPage { get; set; }
  System.Drawing.Size fResolution { get; set; }

  bool Start() ;
  void SignalToStop();
  void WaitForStop();

  event NewFrameCamEventHandler NewFrame;

  void PresentationPage_ControlBuffer(object sender, PresentationControlEventArgs e);
}
```

Výpis 6: Rozhraní pro implementaci zásuvných modulů

Další důležitou podmínkou pro načtení modulu je atribut PresentationWPFAttribute, který musí být umístěn u výkonné třídy v každém zásuvném modulu. Attribut obsahuje informace o typu přenášených dat, interní jméno modulu a jméno, které se bude zobrazovat v nabídce aplikace pro spuštění daného modulu. Implementaci atributu lze vidět v kódu 7.

```
[System.AttributeUsage(System.AttributeTargets.Class)]
public class PresentationWPFAttribute : System.Attribute
{
  private CamInfo camInfo;

  public PresentationWPFAttribute(ECamType camType, string internalName, string
      displayName)
  {
    this.camInfo = new CamInfo(camType, internalName, displayName);
  }

  public CamInfo CamInfo
  {
    get { return camInfo; }
    set { camInfo = value; }
  }
}
```

Výpis 7: Povinný atribut pro výkonnou třídu zásuvného modulu

8.2.4 Silverlight klient

Tento klient je určen pro běh ve webovém prohlížeči. Není nutné instalovat žádné dodatečné aplikace do systému Windows. Stačí přejít na webovou stránku s klientem a přihlásit se do kurzu. Tento klient slouží po sledování webového semináře. V malé míře může účastník do semináře zasahovat pomocí textových zpráv, které jsou zasílány formou chatu mezi všemi účastníky semináře.

Jednou z funkcí, která je pro tohoto klienta dostupná navíc oproti desktopové verzi, je možnost pořizovat snímky z prezentace. Ty se ukládají jako miniatury v panelu na okraji obrazovky. Jednotlivé snímky lze ukládat do souboru ve formátu JPEG. Po dvojkliku na miniaturu snímku v panelu se zobrazí jeho plná velikost na místě s probíhající prezentací. Prezentace se přesune do panelu s miniaturami obrázků, kde je zobrazen živý náhled z probíhající prezentace. Účastník tak stále může sledovat probíhající seminář a při tom pracovat s jednotlivými obrázky.

8.2.4.1 Architektura Silverlight klienta Klient je postaven na technologii Microsoft Silverlight. Tato technologie umožnila běh aplikace v prostředí webového prohlížeče bez nutnosti instalovat dodatečný software do počítače. Implementace tohoto klienta je převzata z bakalářské práce. Zásadní změny jsou v komunikaci tohoto klienta se síti FroceB. Jednotlivé knihovny implementující FBP protokol bylo nutné přepsat na platformu Microsoft Silverlight. Implementace FBP protokolu je v knihovnách se jemnným prostorem ForceB.SL.Communication.

8.3 Implementace IS

Informační systém slouží k evidenci kurzu a administraci uživatelů. Nabízí rozdělení kurzu do jednotlivých kategorií podle zaměření. Implementace tohoto systému byla převzata z bakalářské práce. Databáze systému byla převedena z technologie firmy Oracle na Microsoft SQL Server. S tím souvisela i nová implementace objektově relačního mapování na straně serveru pomoci technologie ADO.NET Entity Framework [27].

9 ForceB na Windows Azure

V této kapitole bude popsána architektura cloud computing [23] platformy Windows Azure. Jsou zde shrnuty možnosti této platformy a popsány některé služby, které společnost Microsoft v rámci svého cloud řešení nabízí. Další část se zabývá implementaci systému ForceB do prostředí cloud a popisuje výhody, které jsou spojeny s využitím této platformy.

9.1 Microsoft Windows Azure

Systém ForceB je už od počátku vyvíjen pro platformu Windows Azure [24, 25]. Jde o cloud computing platformu společnosti Microsoft, která umožňuje běh aplikací hostovaných v datacentrech. Poskytuje celou škálu služeb pro vytváření aplikací, jejichž použití sahá od spotřebitelských webů až po nasazení v podnicích. Spadá do kategorie PaaS neboli Platform as a Service [12]. Zákazník tedy dostává platformu pro aplikace jako službu. PaaS umožňuje zákazníkovi zaměřit se na inovace, namísto toho aby řešil složitou hardwarovou infrastrukturu. Vše co zákazník potřebuje, dostane ve formě služby, za kterou obvykle platí modelem pay as you go. Platí tedy pouze za to, co doopravdy spotřebuje.

Platformu Windows Azure můžeme rozdělit do několika částí:

- **Windows Azure** – Jedná se o operační systém, který slouží pro hostování a správu služeb.
- **SQL Azure** – Relační databázová služba pro prostředí cloudu.
- **Azure Storage** – Několik typů úložného prostoru pro uživatelská data.

9.1.1 Windows Azure

Windows Azure je cloudový operační systém založený na Windows Server 2008 R2 64bit s využitím technologie Hyper-V. Aplikace pro toto prostředí jsou postaveny na .NET technologiích. Právě díky tomu je možné aplikace napsané pro klasické servery lehce přizpůsobit pro běh v cloudu.

Při vytváření aplikace si volíme mezi dvěma rolemi. Každá role je určena pro odlišný typ aplikace. Pro webové aplikace je připravená web role. Na pozadí běží IIS web server, kde můžeme snadno publikovat webové aplikace.

Podporovány jsou programovací jazyky ASP .NET, PHP nebo Java. Právě zde běží IS systému ForceB. Dalším typem role jsou worker role. Primárně je tato role navržena pro běh aplikací na pozadí. Můžeme je přirovnat k systémovým službám OS Windows. Tyto role v systému ForceB reprezentují server a jednotlivé uzly. Pro obě role jsou připraveny API funkce pro přístup k Azure Storage a přístup k SQL Azure. Obě role mohou běžet současně v rámci jedné instance serveru nebo mohou běžet odděleně. V takovém případě je pro každou roli přidělen samostatný virtuální stroj.

9.1.2 SQL Azure

Microsoft nabízí v rámci svého balíčku služeb pro Windows Azure i relační databázi SQL Azure. Právě zde jsou uložena veškerá relační data systému ForceB. Tato databáze je založena na systému SQL Server 2012. Nejedná se ovšem o totožný systém. Najdeme zde několik omezení, která souvisí s implementací databáze v prostředí cloudu. SQL server není vyhrazený jenom pro jednoho klienta, a proto nelze využívat funkce, které by byly rizikem pro bezpečnost, výkonnost a stabilitu celého serveru. Nepodporuje například integraci CLR, zrcadlení databáze, distribuované dotazy, fulltextové vyhledávání a další.

Naopak oproti klasickému SQL serveru má několik výhod. SQL Azure udržuje neustále několik totožných replik databáze. V jednom okamžiku existují minimálně tři repliky databáze a jedna primární. Jedna z replik je navíc umístěna v odlišném datacentru pro případ katastrofy. Pokud selže primární replika, je za primární označena některá záložní a ihned se vytvoří nová záložní replika tak, aby existovaly vždy minimálně tři záložní.

Programátor může se službou SQL Azure pracovat úplně stejným způsobem jako s jinou MS SQL databází. Rozdíl je pouze v connection stringu. Klientská aplikace s databází komunikuje standardním subsystémem ADO.NET [27].

9.1.3 Azure Storage

Tato služba poskytuje různorodé přístupy k datům v prostředí Windows Azure. Služba je určena pro ukládání binárních dat do několika struktur. Výhodou je vysoká škálovatelnost, dostupnost a trvanlivost dat. Data lze ukládat do tří typů uložišť:

- **Bloby** – Jsou určeny pro velké binární data, jako jsou dokumenty, obrázky, atd.
- **Tabulky** – Nerelační databáze k uložení velkých objemů strukturovaných dat.
- **Fronty** – Slouží pro frontu zpráv. Využívá se pro předávání zpráv mezi worker rolemi.

9.2 Implementace systému ForceB na Windows Azure

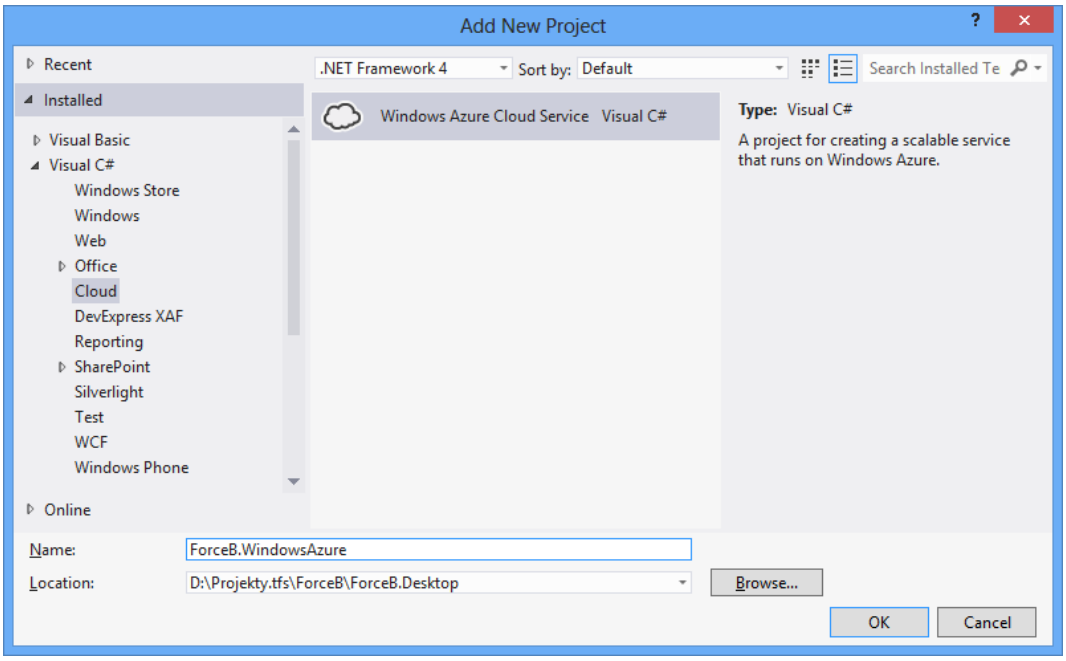
Jako nejvhodnější řešení pro běh serverové částí systému ForceB byl zvolen cloud. K tomuto řešení jsem přistoupil hned z několika důvodů. Windows Azure umožňuje na základě aktuálního vytížení sítě dynamicky navyšovat výpočetní výkon. To se hodí zejména v případech, kdy ke službě přistupují klienti pouze v konkrétní denní dobu. Pro systém ForceB se počítá se špičkou mezi osmou hodinou ranní až čtvrtou hodinou odpolední. Naopak v době večera a noci, bude vytížení systému minimální a není tedy potřeba velkého výpočetního výkonu. Od spotřebovaných systémových prostředků se následně odvíjí i cena, kterou si Microsoft účtuje za své služby.

9.2.1 Nasazení systému ForceB na Windows Azure

Pro nasazení aplikace do prostředí cloudu má Microsoft připraveny nástroje ve vývojovém prostředí Visual Studio 2010 a novější. Proto, aby bylo možné tyto služby

využívat, bylo nutné si Windows Azure aktivovat. To lze provést na webové stránce www.windowsazure.com.

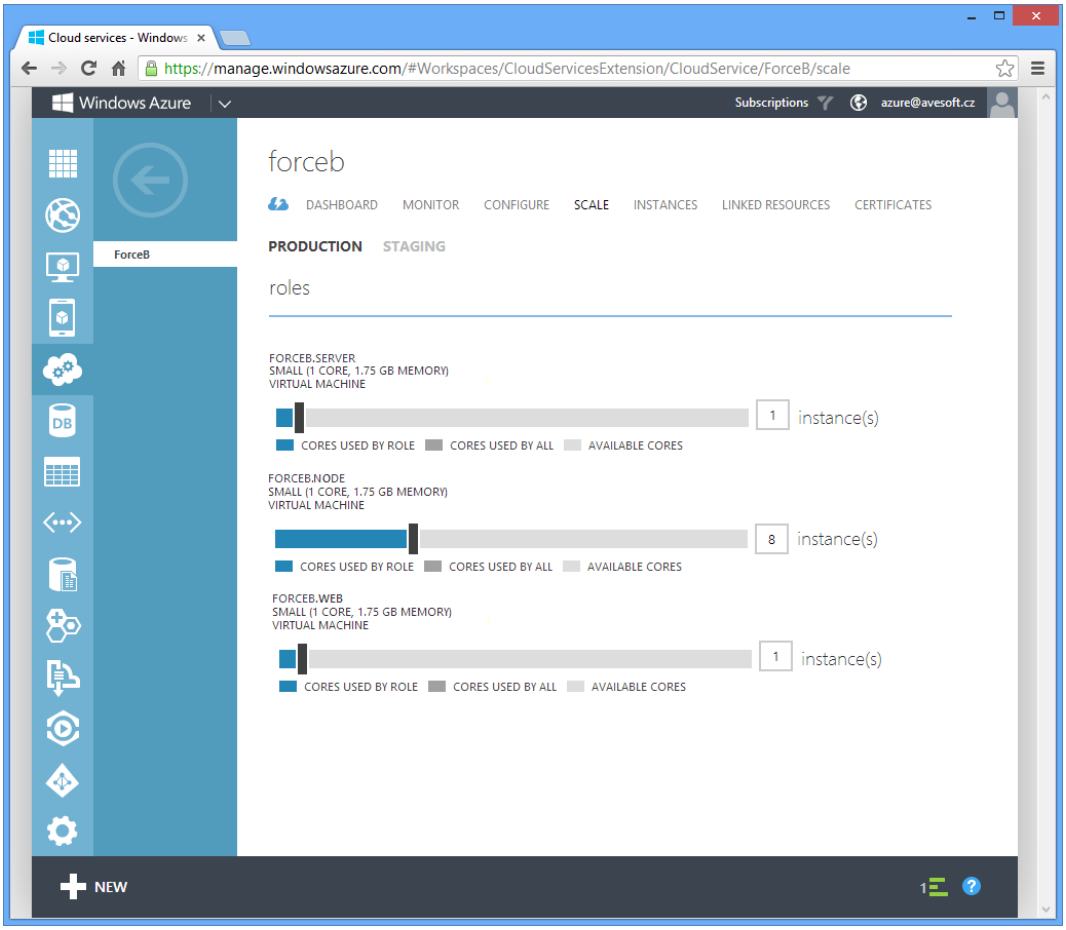
Do solutiony ForceB se přidal nový Windows Azure Cloud Service projekt, viz obrázek 18. Následně bylo nutné specifikovat služby, které chceme v cloud aplikaci použít. Pro účely systému ForceB byly zvoleny dvě worker role. Jedna slouží pro instanci serveru a druhá worker role je určena pro uzly. Worker role s uzly je poté spuštěna v několika instancích. Další roli je webová role, která obsluhuje informační systém FroceB.



Obrázek 18: Založení nového projektu Windows Azure Cloud Service

9.2.2 Práce s worker rolemi a rozložení zátěže

Microsoft Azure umožňuje dynamicky upravovat množství výpočetního výkonu. Vše lze jednoduše nastavit ve webovém prohlížeči, viz obrázek 19. Výkon je možné navyšovat pomoci zvýšení počtu worker rolí nebo navýšením hardwarové konfigurace jednotlivých strojů. Každá další worker role je novým uzlem ve ForceB síti. Ihned po spuštění další instance worker role se tento uzel připojí do ForceB sítě.

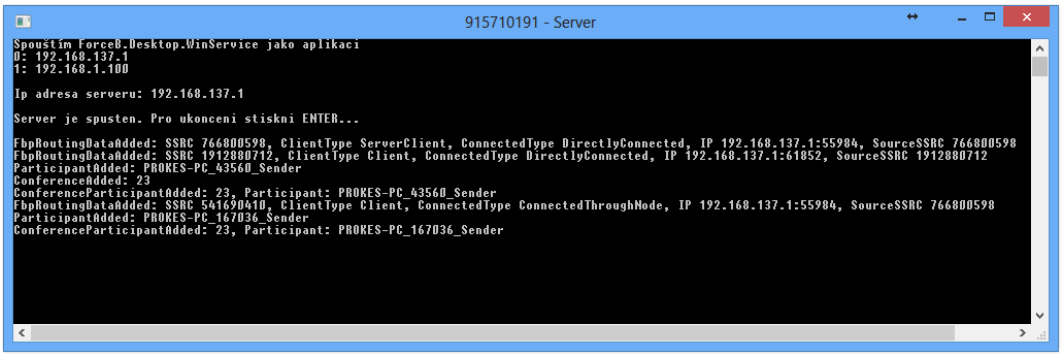


Obrázek 19: Windows Azure Manager

10 Ukázka jednotlivých aplikací systému ForceB

10.1 Aplikační server

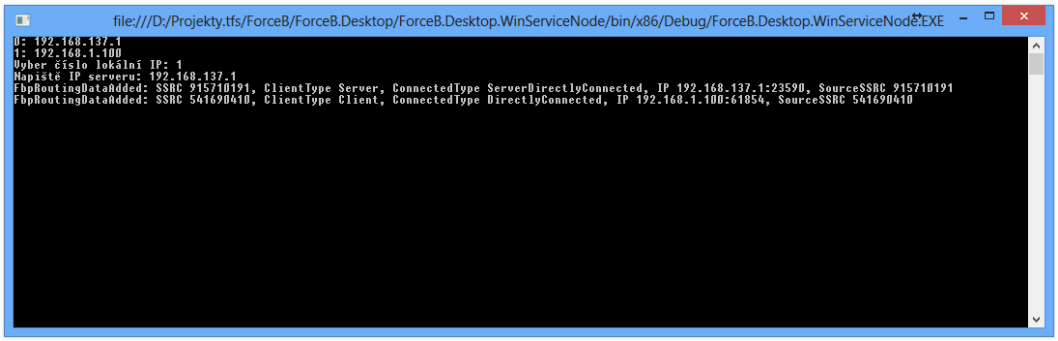
Aplikační server je nejdůležitější část síťové komunikace. Server může být spuštěn v rámci služeb systému Windows nebo jako worker role ve Windows Azure. Na obrázku 20 je server spuštěn v konzoli s výpisem události. Je zde vidět připojení nových klientů do kurzu, založení konference a přidání klienta do této konference.



Obrázek 20: Ukázka aplikačního serveru

10.2 Uzel

Uzly se ve ForceB síti starají o rozložení zátěže v síti. Komunikaci, kterou by musel obsloužit server, je nyní možné nechat na jednotlivých uzlech. S každým novým uzlem je možné připojit desítky nových účastníků. Podobně jako server může být spuštěn i uzle. Může jít o službu systému Windows nebo o worker roli ve Windows Azure. Na obrázku 21 je uzel spuštěn v konzoli s výpisem události. Je zde vidět, připojení k serverové části a připojení nového účastníka k tomuto uzlu.

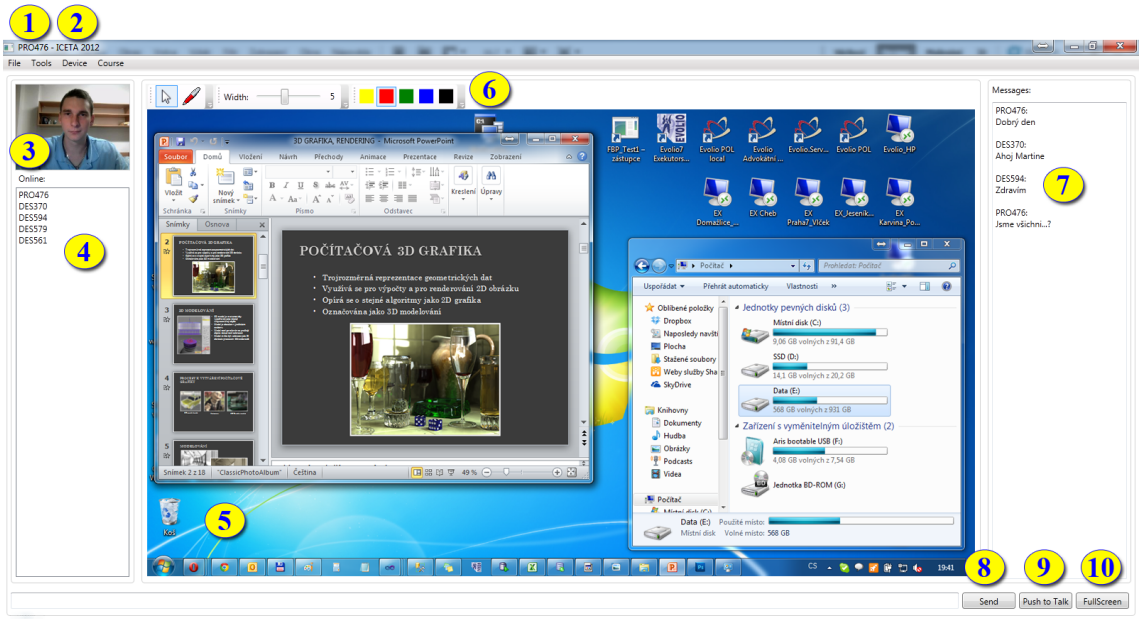


Obrázek 21: Ukázka uzlu

10.3 Desktopový klient

Desktopový klient slouží k pořádání online kurzů a seminářů. Umožňuje posílat textové zprávy, zvuk a obraz z webové kamery ostatním účastníkům kurzu. Dále v závislosti na nainstalovaných zásuvných modulech může vyučující pořádat různé druhy webinářů. Na obrázku 22 je možné vidět probíhající seminář se sdílením pracovní plochy a sdílenou webovou kamerou prezentátora. Zároveň lze vidět, že se kurzu účastní několik klientů. Na obrázku je také vidět nástrojová lišta zásuvného modulu pro sdílení obrazovky. Díky ní může účastník zasahovat interaktivně do výuky. V tomto případě může ovládat vzdálený kurzor nebo pomoci pera kreslit na obrazovku přednášejícího. Další nástroje slouží k zvolení barvy čáry a k nastavení její tloušťky.

10.3.1 Rozvržení aplikace



Obrázek 22: Rozvržení jednotlivých prvků v Desktopové aplikaci

1. **CName** – Jméno přihlášeného uživatele.
2. **ConferenceName** – Jméno kurzu, ve kterém je účastník přihlášen.
3. **WebCam** – Obraz z webové kamery přednášejícího.
4. **OnlineUser** – Seznam přihlášených uživatelů.
5. **Webinar** – Obraz z webového semináře.

6. **ToolBar** – Nástrojová lišta. Každý ze zásuvných modulů může mít odlišný panel nástrojů.
7. **Chat** – Panel s konverzací mezi účastníky a přednášejícím.
8. **Send** – Tlačítko pro odeslání textové zprávy.
9. **PushToTalk** – Tlačítko stiskni a mluv. Po dobu stisknutí tohoto tlačítka může klient mluvit k ostatním účastníkům kurzu.
10. **FullScreen** – Tlačítko pro maximalizaci webového semináře na celou obrazovku.

10.3.2 Zásuvné moduly

Zásuvné moduly souží pro pořádání různých typů webových seminářů. S každým zásuvným modulem dostává aplikace nové funkce a možnosti výuky. Mezi základní zásuvné moduly patří sdílení obrazovky, sdílení okna konkrétní aplikace a sdílení prezentací.

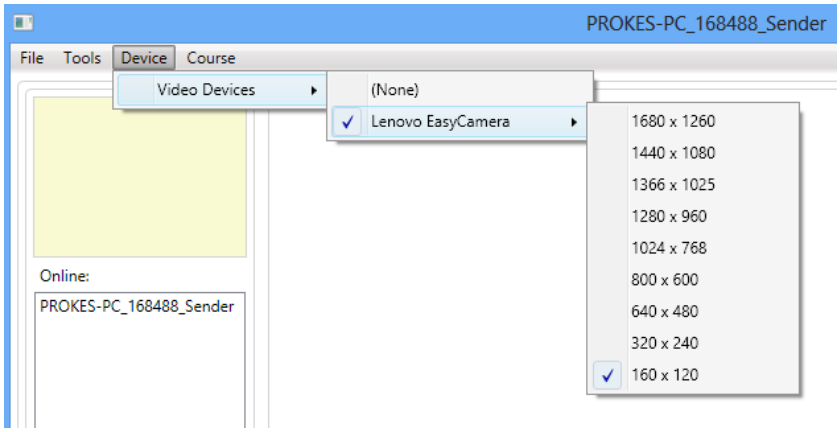
10.3.2.1 Sdílení obrazovky Tento zásuvný modul dovoluje studentům nacházejícím se v kurzu vidět přesně to, co má vyučující na své obrazovce. Snadno tak lze vysvětlit práci v některé aplikaci. Kurz může být například zaměřen na programování. Vyučující zde může objasnit implementaci některého algoritmu ve vývojovém prostředí nebo studenty seznámit se samotným vývojovým prostředím. Studentům lze také umožnit kreslit pomoci štětce přímo na obrazovku vyučujícího. Mohou tak snadno označit místo, ke kterému chtějí znát podrobnější informace. V případě potřeby může vyučující povolit studentům ovládat vlastní počítač na dálku.

10.3.2.2 Sdílení okna konkrétní aplikace Jedná se o velmi podobnou funkci, jako je sdílení obrazovky. V tomto případě však není sdílená celá plocha obrazovky ale jen okno konkrétní aplikace. Studenti tak nevidí další aplikace, které má vyučující spuštěné.

10.3.2.3 Sdílení prezentace Umožňuje sdílet jednotlivé snímky z prezentace vytvořené v aplikaci Microsoft PowerPoint [9]. Vyučující si tak k přednášce může připravit podklady, které mu lépe pomohou vysvětlit problematiku dané látky. Dále může studentům povolit kreslit nebo psát přímo na plátno prezentace pomoci vestavěných nástrojů. Tento plugin má navíc oproti ostatním výhodu v datové náročnosti na šířku přenosového pásma. Přenáší se pouze změna a nikoliv obraz s vysokým množstvím snímku za sekundu.

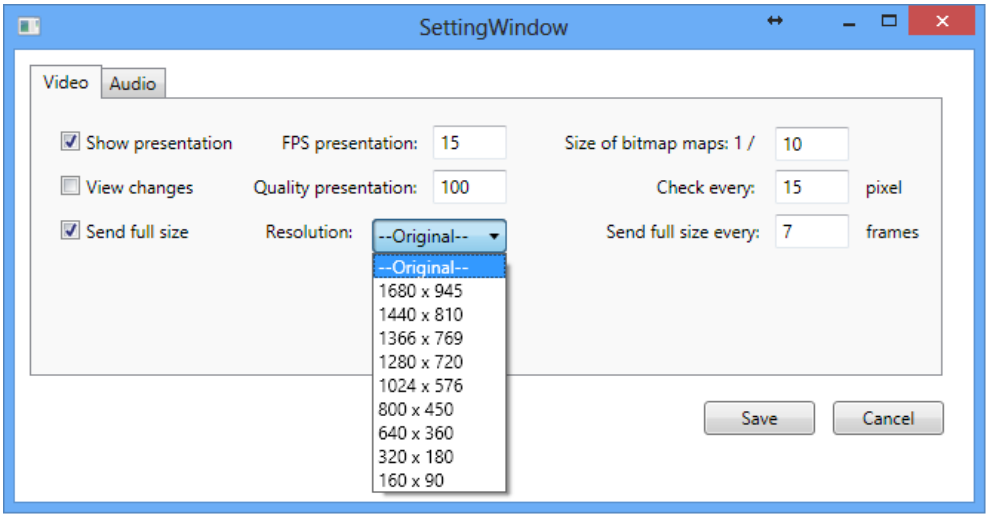
10.3.3 Nastavení

10.3.3.1 Výběr webové kamery Uživatel má možnost si z nastavení vybrat webovou kameru, použitou pro sdílení videa. Zároveň může zvolit vhodné rozlišení této kamery. Tato možnost se hodí v případě, že je pro webový seminář použita externí webová kamera.



Obrázek 23: Nastavení webové kamery a jejího rozlišení

10.3.3.2 Nastavení kvality videa webového semináře Účastník vystupující jako prezentátor má možnost nastavit kompresi výstupního obrazu z webového semináře. Může nastavovat rozlišení videa a konfigurovat čtvercovou kompresi. Na obrázku 24 je vidět okno pro konfiguraci těchto možností.



Obrázek 24: Dialogové okno pro nastavení videa

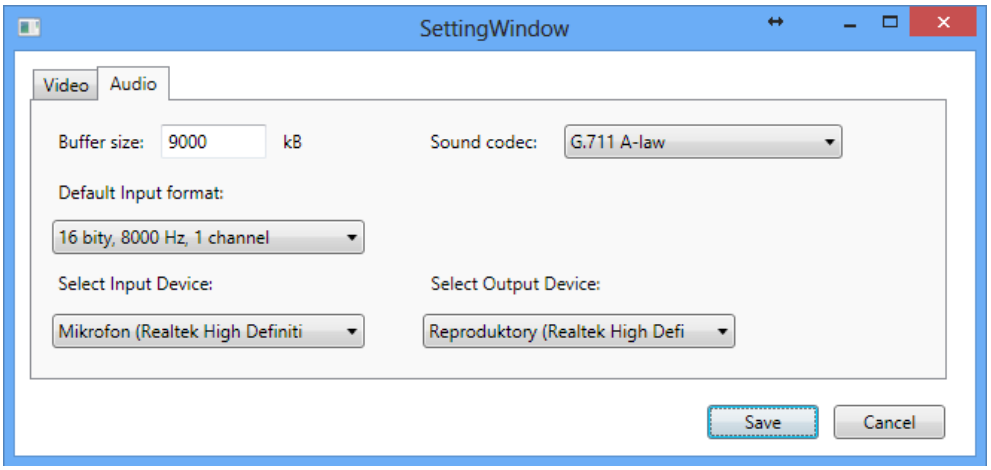
Jednotlivé volby v nastavení udávají:

- **Show presentation** – Povolí nebo zakáže zobrazovat obraz z webového semináře.
- **View changes** – Pokud je volba povolena, budou se v jednotlivých snímcích zobrazovat červené tečky na místech, kde došlo oproti předešlému snímku ke změně.

- **Send full size** – Po zaškrtnutí této volby se vypne čtvercová komprese a začne se odesílat celý obraz.
- **FPS presentation** – Nastavuje počet snímků za sekundu pro webový seminář.
- **Quality presentation** – Úroveň JPEG komprese (0 = nejnižší kvalita, 100 = nejvyšší kvalita).
- **Resolution** – Udává rozlišení videa, které se bude zasílat z webového semináře.
- **Size of bitmap maps** – Velikost bitmap v poměru k velikosti snímané plochy. Čím je hodnota nižší, tím bude velikost bitmap větší a naopak.
- **Check every x pixel** – Nastavuje velikost rozteče v pixelech, po kterých se hledá změna ve snímku. Kontrolovat každý pixel snímku by bylo velmi náročné na procesorový výkon počítače.
- **Send full size every x frames** – Nastavuje interval, v jakém se zasílá plná velikost snímků.

10.3.3.3 Nastavení kvality audia webového semináře

Každý účastník může v závislosti na své šířce přenosového pásma nastavit kvalitu odesílaného zvuku. Toto nastavení se týká zvuku z webového semináře a i zvuku odesílaného po stisknutí tlačítka PushToTalk.



Obrázek 25: Dialogové okno pro nastavení audia

Jednotlivé volby v nastavení udávají:

- **Buffer size x kB** – Velikost vyrovnávací paměti pro audio.
- **Sound codec** – Kodek pomocí kterého bude komprimován zvuk.
- **Default Input format** – Kvalitu vstupu z mikrofonu. Lze nastavit btitovou hloubku, frekvenci a počet kanálů.

- **Select Input Device** – Nastavení vstupního zařízení.
- **Select Output Device** – Nastavení výstupního zařízení.

10.3.4 Systémové a hardwarové požadavky

10.3.4.1 HW vybavení

- **Procesor** - Intel Dual core procesor (případně alternativní AMD).
- **Operační paměť** - 1 GB RAM.
- **Disk**- 100 MB.
- **Rozlišení obrazovky** - minimálně 1024x768 bodů.
- **Mikrofon.**
- **Webkamera** - nepovinné.

10.3.4.2 SW vybavení

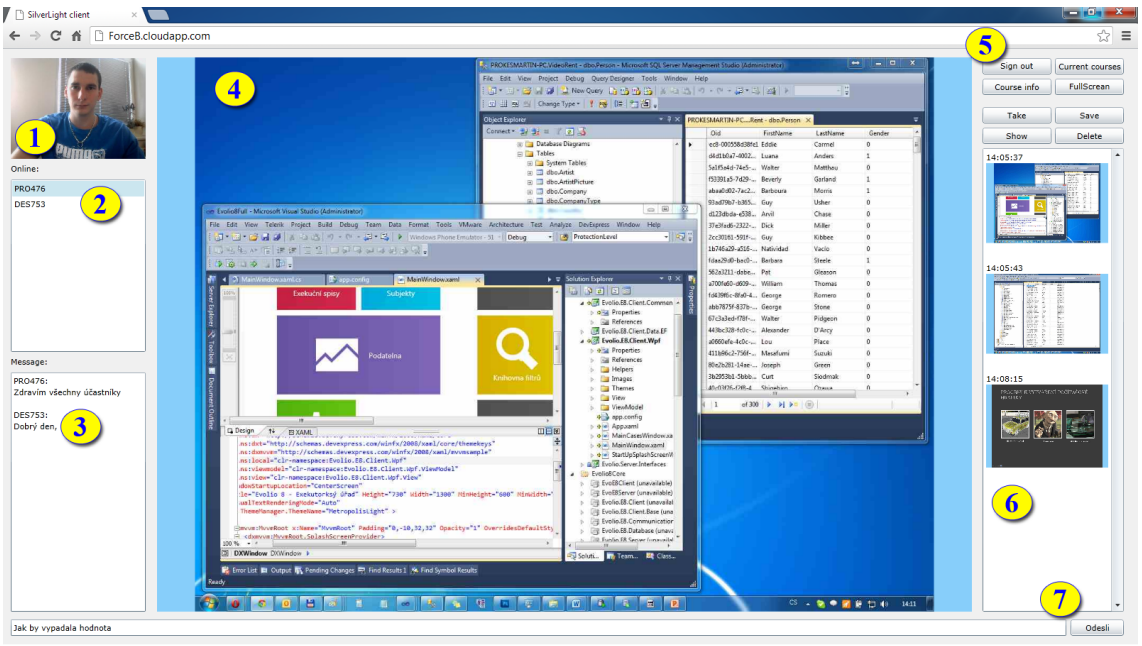
- Operační systém: MS Windows Vista, Windows 7.
- Microsoft .NET Framework 3.5.
- Doporučen Microsoft Office 2003/2007.

10.3.4.3 Internetová konektivita

- **Download** - 500-600kB/s.
- **Upload** - 500-600kB/s (v případě pořádaní kurzu) 30kB/s (v případě sledování kurzu).

10.4 Silverlight klient

Jde o klienta určeného pro webové prohlížeče s podporou technologie Microsoft Silverlight [20]. Pokud se student kurzu účastní jako pozorovatel, nemusí nic stahovat ani instalovat. Stačí mu pouze přejít na webovou stránku klienta a přihlásit se. Silverlight klient nabídne oproti desktopové verzi některé funkce navíc. Jednou z nich je možnost obraz z prezentace přiblížit nebo oddálit. Další je schopnost pořizovat snímky z prezentace. Ty se ukládají jako miniatury v panelu na okraji obrazovky. Tyto snímky lze následně zobrazit v prostředí klienta nebo je uložit do souboru ve formátu JPEG.



Obrázek 26: Rozvržení jednotlivých prvků v Silverlight aplikaci

10.4.1 Rozvržení aplikace

1. **WebCam** – Obraz z webové kamery přednášejícího.
2. **OnlineUser** – Seznam přihlášených uživatelů.
3. **Chat** – Panel s konverzací mezi účastníky a přednášejícím.
4. **Webinar** – Obraz z webového semináře.
5. **ToolBar** – Nástrojová lišta.
6. **PicturePanel** – Panel s miniaturami vyfocených obrázku ze semináře.
7. **Send** – Tlačítko pro odeslání textové zprávy.

10.4.2 Systémové a hardwarové požadavky

10.4.2.1 HW vybavení

- **Procesor** - Intel Dual core procesor (případně alternativní AMD).
- **Operační paměť** - 512 MB RAM.
- **Rozlišení obrazovky** - minimálně 1024x768 bodů.

10.4.2.2 SW vybavení

- Operační systém - MS Windows XP, Vista, Windows 7.
- Microsoft Silverlight.

10.4.2.3 Internetová konektivita

- **Download** - 500-600kB/s.
- **Upload** - 30kB/s.

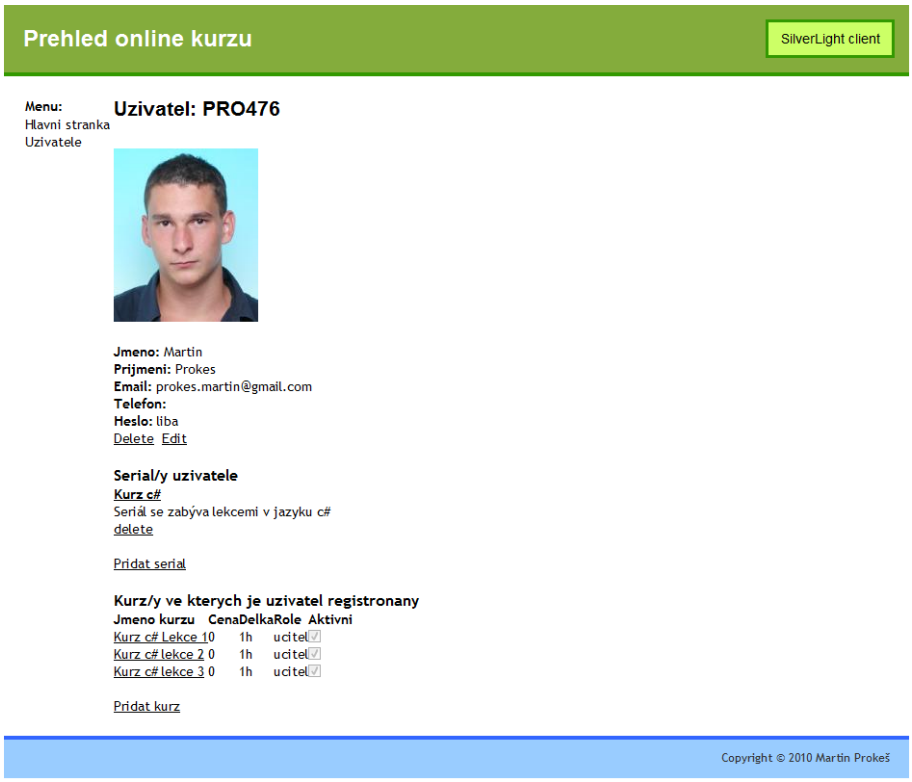
10.5 Webový informační systém

Slouží k evidenci kurzu a uživatelů. Nabízí rozdělení kurzu do jednotlivých kategorií podle zaměření. Kurzy, které jsou rozděleny do několika částí (například, pokud daná problematika obsáhne několik lekcí) je možné shlukovat do seriálů.

Dále zprostředkovává studentům informace o jednotlivých kurzech. Jsou zde obsaženy údaje o termínech, délce kurzu, ceně, motivaci, obsahu a cílové skupině. Ke kurzu je možné přikládat i doprovodné dokumenty. Ty mohou obsahovat dodatečné informace k probírané látce.

Pokud se počítá s vysokým zájmem o určitý kurz, existuje zde možnost registrace do kurzu. Student si tak rezervuje své místo a má jistotu, že se do kurzu v době konání přihlásí. Vyučující zde mohou registrovat nové kurzy, seriály a provádět nad nimi administraci.

10.5.1 Ukázka webového rozhraní



Obrázek 27: Ukázka webového rozhraní

11 Závěr

Diplomová práce popisuje vývoj systému pro pořádání webových seminářů v prostředí univerzitní sítě a internetu. Zabývá se všemi aspekty souvisejícími s problematikou webinářů v prostředí internetu. Důraz je kladen především na přenos dat a s tím související vývoj vlastního komunikačního protokolu FBP.

Cílem práce bylo navrhnout a implementovat systém pro pořádání webinářů, který by umožnil pořádat kurzy až pro stovky účastníků. Snahou bylo zvolit řešení postavené na nových technologiích, které bude snadným způsobem dále rozšiřitelné o podporu nových funkcí pomoci zásuvných modulů. Mezi nejdůležitější požadavky patřila možnost pořádat semináře pro studenty bez nutnosti instalace dodatečného softwaru.

Textová forma diplomové práce pak měla za cíl popsat architekturu vlastního komunikačního protokolu FBP, popsat implementaci celého systému a představit aplikace, které se podílejí na běhu celého systému.

Práce na projektu ForceB pro mne měla obrovský přínos. Získal jsem cenné zkušenosti s vývojem aplikací na různých platformách a jejich nasazením do prostředí cloudu. Rozšířil jsem si taky znalosti z oblastí počítačových sítí.

Obrovskou zkušeností pro mne byla možnost účastnit se mezinárodní konference ICETA 2012. Měl jsem zde možnost seznámit se s různými projekty s mezinárodním rozsahem. Zároveň jsem získal cennou zpětnou vazbu od účastníků konference, která byla inspirací pro zaměření se na konkrétní části systému. Další velkou zkušeností pro mne byla účast na semináři ORGANON, kde jsem měl možnost prezentovat projekt ForceB ostatním účastníkům semináře.

Následující vývoj se bude ubírat nasazením tohoto systému do produkčního prostředí. Plánován je i vývoj klientů pro další platformy jako je Windows Phone nebo Android.

12 Reference

[1] *PROKEŠ Martin, Nástroj pro sdílenou pracovní plochu a vzdálenou výuku.* Ostrava, 2011. Bakalářská práce. Vysoká škola báňská - Technická univerzita Ostrava.

[2] *International Conference on Emerging eLearning Technologies and Applications.* Http://iceta.sk/ [online]. c2013 [cit. 2013-05-06]. Dostupné z: http://iceta.sk/

[3] *Tool for desktop sharing and remote teaching - ForceB.* Http://ieeexplore.ieee.org [online]. c2013 [cit. 2013-05-06]. Dostupné z: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6418615&contentType=Conference+Publications&searchField%3DSearch_All%26queryText%3DForceB

[4] *International Conference on Emerging eLearning Technologies and Applications.* Http://www.ieee.org [online]. c2013 [cit. 2013-05-06]. Dostupné z: http://www.ieee.org/index.html

[5] *ORGANON - Semináře o výuce logiky.* Http://home.zcu.cz [online]. c2013 [cit. 2013-05-06]. Dostupné z: http://home.zcu.cz/~ldostal/organon8/?id=oprojektu

[6] *HUJKA, Petr. Real - Time Transport Protocol a aplikační rozhraní RTP Java Media Framework.*[Online] Http://www.elektrorevue.cz [online]. 27.5.2003 [cit. 2013-05-06]. Dostupné z: http://www.elektrorevue.cz/clanky/03018/index.html.

[7] *IETF. RTP: A Transport Protocol for Real-Time Applications.* Http://www.ietf.org [online]. 2003 [cit. 2013-05-06]. Dostupné z: http://www.ietf.org/rfc/rfc3550.txt.

[8] *Webináře* [online]. c2011 [cit. 2013-05-06]. O webinářích. Dostupné z WWW: <http://www.webinare.cz/o-webinarich.aspx.

[9] *VEČERKA, ARNOŠT. KOMPRESE DAT* [online]. Olomouc : [s.n.], 2008 [cit. 2013-05-06]. Dostupné z WWW: http://phoenix.inf.upol.cz/esf/ucebni/komprese.pdf.

[10] *KOMENDA, Martin . Autorský nástroj Ozvučená prezentace* [online]. [s.l.], 2007. 51 s. Bakalářská práce. Masarykova univerzita. Dostupné z WWW: http://is.muni.cz/th/98951/fi_b/.

[11] *GRACÍK, Martin. Sdílená pracovní plocha* [online]. [s.l.], 2008. 22 s. Bakalářská práce. Masarykova univerzita. Dostupné z WWW: http://is.muni.cz/dok/rfmgr.pl?furl=%2Fth%2F143087%2Ffi_b%2F;info=.

[12] *Platform as a Service – When it comes to the cloud, PaaS is the point.* Http://vlele.wordpress.com [online]. 2010 [cit. 2013-05-06]. Dostupné z: http://vlele.wordpress.com/2010/12/07/platform-as-a-service-when-it-comes-t

[13] *Software as a Service: Strategic Backgrounder : Software and Information Industry Association* [online]. Washington, D.C : [s.n.], 2001 [cit. 2013-05-06]. Dostupné z WWW: http://www.siia.net/estore/pubs/SSB-01.pdf.

[14] *What Is Unicast IPv4 Routing?.* Http://technet.microsoft.com [online]. 2003 [cit. 2013-05-06]. Dostupné z: http://technet.microsoft.com/de-de/library/cc736574(WS.10).aspx

[15] *Multicast over TCP/IP HOWTO.* Http://www.tldp.org [online]. 2003 [cit. 2013-05-06]. Dostupné z: http://www.tldp.org/HOWTO/Multicast-HOWTO.html

[16] *Adobe Connect* [online]. c2013 [cit. 2013-05-06]. Dostupné z WWW: <http://www.adobe.com/products/adobeconnect.html.

[17] *WebEx Meetings. Http://www.webex.com/* [online]. c2013 [cit. 2013-05-06]. Dostupné z: http://www.webex.com/

[18] *GoToWebinar* [online]. c2013 [cit. 2013-05-06]. Dostupné z WWW: http://www.gotomeeting.com/fec/webinar.

[19] *Microsoft ASP.net* [online]. c2013 [cit. 2013-05-06]. Dostupné z WWW: http://www.asp.net/.

[20] *Microsoft /web* [online]. c2013 [cit. 2013-05-06]. Microsoft Silverlight. Dostupné z WWW: <http://www.microsoft.com/cze/web/silverlight/.

[21] *Microsoft Visual Studio* [online]. c2013 [cit. 2013-05-06]. Dostupné z WWW: http://www.microsoft.com/cze/msdn/vstudio/

[22] *BĚHÁLEK, Marek. Programovací jazyk C#* [online]. 2007 [cit. 2013-05-06]. Dostupné z WWW: http://www.cs.vsb.cz/behalek/vyuka/pcsharp/text/index.html.

[23] *VELTE, Anthony T. Cloud computing: praktický průvodce* Vyd. 1, Brno: Computer Press, 2011, 344 s. ISBN 978-80-251-3333-0.

[24] *REDKAR, Tejaswi a Tony GUIDICI. Windows Azure platform.* New ed, New York: Apress, 2009. ISBN 14-302-2479-7.

[25] *KRISHNAN, Sriram Programming Windows Azure.* 1st ed., Sebastopol, CA: O'Reilly, 2010, xix, 345 p. ISBN 978-059-6801-977.

[26] *Dalibor Kačmář, Přednáška Windows Azure*, In: Cs.vsb.cz [online]. 2011 [cit. 2013-05-06]. Dostupné z:
http://www.cs.vsb.cz/katedra/zaznamy-z-akci/prednaska-windows-azure.aspx.

[27] *JUŘEK Michael, ADO.NET ENTITIES: 1. K ČEMU JE TO DOBRÉ?* http://blog.vyvojar.cz/ [online]. 2008 [cit. 2013-05-06]. Dostupné z:
http://blog.vyvojar.cz/mjurek/archive/2008/04/07/ADO.NET-Entities_3A00_-1.-cast-_2D00_-K-cemu-je-to-dobre.aspx

[28] *Microsoft PowerPoint.* Http://office.microsoft.com [online]. 2013 [cit. 2013-05-06]. Dostupné z:
http://office.microsoft.com/en-us/microsoft-powerpoint-slide-presentation-s
aspx

[29] *Introducing Windows Presentation Foundation.* Http://msdn.microsoft.com [online]. 2013 [cit. 2013-05-06]. Dostupné z:
http://msdn.microsoft.com/en-us/library/aa663364.aspx

[30] *Bandwidth.* Http://www.techterms.com [online]. 2012 [cit. 2013-05-06]. Dostupné z:
http://www.techterms.com/definition/bandwidth

[31] *Implementing Quality of Service Policies with DSCP.* Http://www.cisco.com [online]. 2008 [cit. 2013-05-06]. Dostupné z:
http://www.cisco.com/en/US/tech/tk543/tk757/technologies_tech_note09186a00800949f2.shtml

[32] *CESNET.* Http://www.cesnet.cz/ [online]. 2013 [cit. 2013-05-06]. Dostupné z:
http://www.cesnet.cz/sdruzeni/

[33] *Topologie počítačových sítí.* Http://home.zcu.cz/ [online]. c2013 [cit. 2013-05-06]. Dostupné z:
http://home.zcu.cz/~topinkov/druhy.html

[34] *TCP a UDP.* Http://www.earchiv.cz [online]. c2013 [cit. 2013-05-06]. Dostupné z:
http://www.earchiv.cz/anovinky/ai1864.php3

[35] *Network Broadcasting and Multicasting.* Http://www.comptechdoc.org [online]. c2013 [cit. 2013-05-06]. Dostupné z:
http://www.comptechdoc.org/independent/networking/guide/netbroadcasting.html

[36] *Microsoft SQL Server.* Http://www.microsoft.com [online]. c2013 [cit. 2013-05-06]. Dostupné z:
http://www.microsoft.com/en-us/sqlserver/default.aspx

A Obsah přiloženého CD

Na přiloženém CD naleznete tyto složky:

- Diplomová práce - text diplomové práce ve formátu pdf